```
; ****************************************************************************
; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.3
; ----------------------------------------------------------------------------
; Last Update: 24/01/2021
; ----------------------------------------------------------------------------
; 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[EA6D]                   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[E66D]                   mov   [mem_1m_1k], cx
00000017 8916[E86D]                   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[E86D]                     mov    si, [mem_16m_64k]
00000076 8936[E10E]                     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[5409]                       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[2543]                       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[E30E]                     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[E86D]                     mov    di, [mem_16m_64k]
000000DD 893E[E10E]                     mov    [real_mem_16m_64k], di

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

000000E5 8936[E86D]                     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
```

```
000000FF FE06[ED6D]          <1>       inc    byte [hdc]   ; count of hard disks (EDD present)
00000103 8816[EC6D]          <1>         mov     [last_drv], dl  ; last hard disk number
00000107 BB[706D]            <1>       mov    bx, hd0_type - 80h
0000010A 01D3                <1>       add    bx, dx
0000010C 880F                <1>       mov    [bx], cl ; Interface support bit map in CX
                             <1>                       ; Bit 0 - 1, Fixed disk access subset ready
                             <1>                       ; Bit 1 - 1, Drv locking and ejecting ready
                             <1>                       ; Bit 2 - 1, Enhanced Disk Drive Support
                             <1>                            ;            (EDD) ready (DPTE ready)
                             <1>                       ; Bit 3 - 1, 64bit extensions are present
                             <1>                                ;            (EDD-3)
                             <1>                       ; Bit 4 to 15 - 0, Reserved
0000010E 80FA83              <1>       cmp    dl, 83h       ; drive number < 83h
00000111 72DB                <1>       jb     short L1
                             <1> L2:
                             <1>       ; 23/11/2014
                             <1>       ; 19/11/2014
00000113 30D2                <1>       xor    dl, dl  ; 0
                             <1>       ; 04/02/2016 (esi -> si)
00000115 BE[EE6D]            <1>       mov    si, fd0_type
                             <1> L3:
                             <1>       ; 14/01/2015
00000118 8816[EB6D]          <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[EE6D]          <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[ED6D]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[EB6D]          <1>         mov     [drv], dl
00000143 8816[EC6D]          <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[ED6D]          <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[EB6D]          <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[EB6D]             <1>       inc    byte [drv]
000001A9 BB[706D]               <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[ED6D]               <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[EB6D]             <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[EB6D]             <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[706D]             <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[EE6D]             <1>       sub    bx, hd0_type - 2 ; 15/01/2015
                                <1>       ;add   bx, drv.status
                                <1>       ;mov   [bx], al
                                <1>       ; 29/08/2020
00000202 8887[3A6E]             <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[1E6E]             <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[206E]             <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[706D]00   <1>       mov    byte [bx+hd0_type-80h], 0
000002D4 808F[BC6D]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[BC6D]FE   <1>       and    byte [bx+drv.status-7Eh], 0FEh
                        <1>       ; 'diskio' procedure will check this bit !
                        <1> L11b:
000002EA 3A16[EC6D]     <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[EB6D]         <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[3A6E]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[F46D]         <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[026E]         <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[106E]         <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[1E6E]         <1>         mov     [bx+drv.size], ax
00000347 8997[206E]         <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[586D]                   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 B9F9640000                     mov  ecx, (bss_end - bss_start)/4
                                        ;shr  ecx, 2 ; bss section is already aligned for double words
000003BE BF[A67E0100]                   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[E66D0000]               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[98810100]                 mov    [free_pages], ecx
000003DF 668B15[E86D0000]               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[98810100]         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[94810100]           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[A8810100]         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[90810100]         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[A0810100]           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[A8810100]       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[98810100]         sub     [free_pages], ecx ; 07/11/2014
                        mi_8:
```

```
000004A2 0FB307                         btr     [edi], eax      ; clear bit 0 to bit x (1 to 31)
                                        ;dec    bl
000004A5 FEC9                           dec     cl
000004A7 7404                           jz      short mi_9
000004A9 FEC0                           inc     al
000004AB EBF5                           jmp     short mi_8
                                mi_9:
                                        ;
                                        ; Reset Kernel's Page Dir. and Page Table bits in M.A.T.
                                        ;             (allocate pages for system page tables)
                                        ;
                                        ; edx = MEM_ALLOC_TBL
000004AD 8B0D[94810100]                 mov     ecx, [memory_size] ; memory size in pages (PTEs)
000004B3 81C1FF030000                   add     ecx, 1023       ; round up (1024 PTEs per table)
000004B9 C1E90A                         shr     ecx, 10                 ; convert memory page count to
                                                                ; page table count (PDE count)
                                        ;
000004BC 51                             push    ecx             ; (**) PDE count (<= 1024)
                                        ;
000004BD 41                             inc     ecx             ; +1 for kernel page directory
                                        ;
000004BE 290D[98810100]                 sub     [free_pages], ecx ; 07/11/2014
                                        ;
000004C4 8B35[90810100]                 mov     esi, [k_page_dir] ; Kernel's Page Directory address
000004CA C1EE0C                         shr     esi, 12                 ; convert to page number
                                mi_10:
000004CD 89F0                           mov     eax, esi        ; allocation bit offset
000004CF 89C3                           mov     ebx, eax
000004D1 C1EB03                         shr     ebx, 3          ; convert to alloc. byte offset
000004D4 80E3FC                         and     bl,  0FCh       ; clear bit 0 and bit 1
                                                                ;   to align on dword boundary
000004D7 83E01F                         and     eax, 31                 ; set allocation bit position
                                                                ;  (bit 0 to bit 31)
                                        ;
000004DA 01D3                           add     ebx, edx        ; offset in M.A.T. + M.A.T. address
                                        ;
000004DC 0FB303                         btr     [ebx], eax      ; reset relevant bit (0 to 31)
                                        ;
000004DF 46                             inc     esi             ; next page table
000004E0 E2EB                           loop    mi_10           ; allocate next kernel page table
                                                                ; (ecx = page table count + 1)
                                        ;
000004E2 59                             pop     ecx             ; (**) PDE count (= pg. tbl. count)
                                        ;
                                        ; Initialize Kernel Page Directory and Kernel Page Tables
                                        ;
                                        ; Initialize Kernel's Page Directory
000004E3 8B3D[90810100]                 mov     edi, [k_page_dir]
000004E9 89F8                           mov     eax, edi
000004EB 0C03                           or      al, PDE_A_PRESENT + PDE_A_WRITE
                                                                ; supervisor + read&write + present
000004ED 89CA                           mov     edx, ecx        ; (**) PDE count (= pg. tbl. count)
                                mi_11:
000004EF 0500100000                     add     eax, 4096       ; Add page size (PGSZ)
                                                                  ; EAX points to next page table
000004F4 AB                             stosd
000004F5 E2F8                           loop    mi_11
000004F7 29C0                           sub     eax, eax        ; Empty PDE
000004F9 66B90004                       mov     cx, 1024        ; Entry count (PGSZ/4)
000004FD 29D1                           sub     ecx, edx
000004FF 7402                           jz      short mi_12
00000501 F3AB                           rep     stosd           ; clear remain (empty) PDEs
                                        ;
                                        ; Initialization of Kernel's Page Directory is OK, here.
                                mi_12:
                                        ; Initialize Kernel's Page Tables
                                        ;
                                        ; (EDI points to address of page table 0)
                                        ; eax = 0
00000503 8B0D[94810100]                 mov     ecx, [memory_size] ; memory size in pages
00000509 89CA                           mov     edx, ecx        ; (***)
0000050B B003                           mov     al, PTE_A_PRESENT + PTE_A_WRITE
                                                                ; supervisor + read&write + present
                                mi_13:
0000050D AB                             stosd
0000050E 0500100000                     add     eax, 4096
00000513 E2F8                           loop    mi_13
00000515 6681E2FF03                     and     dx, 1023        ; (***)
0000051A 740B                           jz      short mi_14
0000051C 66B90004                       mov     cx, 1024
00000520 6629D1                         sub     cx, dx          ; from dx (<= 1023) to 1024
00000523 31C0                           xor     eax, eax
00000525 F3AB                           rep     stosd           ; clear remain (empty) PTEs
                                                                ; of the last page table
                                mi_14:
                                        ;  Initialization of Kernel's Page Tables is OK, here.
                                        ;
00000527 89F8                           mov     eax, edi        ; end of the last page table page
                                                                  ; (beginging of user space pages)
00000529 C1E80F                         shr     eax, 15                 ; convert to M.A.T. byte offset
0000052C 24FC                           and     al, 0FCh        ; clear bit 0 and bit 1 for
                                                                ; aligning on dword boundary

0000052E A3[A4810100]                   mov     [first_page], eax
00000533 A3[9C810100]                   mov     [next_page], eax ; The first free page pointer
                                                                  ; for user programs
                                                                  ; (Offset in Mem. Alloc. Tbl.)
                                        ;
                                        ; Linear/FLAT (1 to 1) memory paging for the kernel is OK, here.
                                        ;

                                        ; Enable paging
                                        ;
00000538 A1[90810100]                     mov     eax, [k_page_dir]
0000053D 0F22D8                         mov     cr3, eax
```

```
00000540 0F20C0                     mov    eax, cr0
00000543 0D00000080                 or     eax, 80000000h      ; set paging bit (bit 31)
00000548 0F22C0                     mov    cr0, eax
                                      ;jmp    KCODE:StartPMP

0000054B EA                         db 0EAh             ; Opcode for far jump
0000054C [52050000]                   dd StartPMP             ; 32 bit offset
00000550 0800                       dw KCODE            ; kernel code segment descriptor

                                StartPMP:
                                      ; 06/11//2014
                                      ; Clear video page 0
                                      ;
                                      ; Temporary Code
                                      ;
00000552 B9E8030000                   mov    ecx, 80*25/2
00000557 BF00800B00                   mov    edi, 0B8000h
                                      ; 30/01/2016
                                      ;xor   eax, eax     ; black background, black fore color
0000055C B800070007                   mov    eax, 07000700h  ; black background, light gray fore color
00000561 F3AB                         rep    stosd

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

                                      ; 29/01/2016 (TRDOS 386 = TRDOS v2.0)
00000563 BE[02450100]                 mov    esi, starting_msg
                                      ;; 14/08/2015 (kernel version message will appear
                                      ;;            when protected mode and paging is enabled)
00000568 BF00800B00                   mov    edi, 0B8000h ; 27/08/2014

                                      ; 30/11/2020
                                      ; 14/11/2020 (TRDOS 386 v2.0.3)
                                      ;cmp  byte [vbe3], 3 ; 03h
                                      ;jne  short pkv_1
                                      ;;mov  ah, 0Bh ; Black background, light cyan forecolor
                                      ;; Light red TRDOS 386 version text shows VBE3 is ready !
                                      ;mov  ah, 0Ch ; Black background, light red forecolor
                                      ;jmp  short pkv_2
                                ;pkv_1:
0000056D B40A                         mov    ah, 0Ah ; Black background, light green forecolor
                                ;pkv_2:
                                      ; 20/08/2014
0000056F E8D6030000                   call   printk

                                      ; 'UNIX v7/x86' source code by Robert Nordier (1999)
                                      ; // Set IRQ offsets
                                      ;
                                      ;  Linux (v0.12) source code by Linus Torvalds (1991)
                                      ;
                                                                  ;; ICW1
00000574 B011                         mov    al, 11h                  ; Initialization sequence
00000576 E620                         out    20h, al                  ;      8259A-1
                                      ; jmp  $+2
00000578 E6A0                         out    0A0h, al           ;      8259A-2
                                                                  ;; ICW2
0000057A B020                         mov    al, 20h                  ; Start of hardware ints (20h)
0000057C E621                         out    21h, al                  ;      for 8259A-1
                                      ; jmp  $+2
0000057E B028                         mov    al, 28h                  ; Start of hardware ints (28h)
00000580 E6A1                         out    0A1h, al           ;      for 8259A-2
                                                                  ;
00000582 B004                         mov    al, 04h                  ;; ICW3
00000584 E621                         out    21h, al                  ;      IRQ2 of 8259A-1 (master)
                                      ; jmp  $+2
00000586 B002                         mov    al, 02h                  ;      is 8259A-2 (slave)
00000588 E6A1                         out    0A1h, al           ;
                                                                  ;; ICW4
0000058A B001                         mov    al, 01h                  ;
0000058C E621                         out    21h, al                  ;      8086 mode, normal EOI
                                      ; jmp  $+2
0000058E E6A1                         out    0A1h, al           ;      for both chips.

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

                                      ; 02/04/2015
                                      ; 26/03/2015 System call (INT 30h) modification
                                      ;  DPL = 3 (Interrupt service routine can be called from user mode)
                                      ;
                                      ;; Linux (v0.12) source code by Linus Torvalds (1991)
                                      ;  setup_idt:
                                      ;
                                        ;; 16/02/2015
                                      ;;mov     dword [DISKETTE_INT], fdc_int ; IRQ 6 handler
                                      ; 21/08/2014 (timer_int)
00000590 BE[A0410100]                 mov    esi, ilist
00000595 8D3D[A87E0100]               lea    edi, [idt]
                                      ; 26/03/2015
0000059B B930000000                   mov    ecx, 48             ; 48 hardware interrupts (INT 0 to INT 2Fh)
                                      ; 02/04/2015
000005A0 BB00000800                   mov    ebx,  80000h
                                rp_sidt1:
000005A5 AD                           lodsd
000005A6 89C2                         mov    edx, eax
000005A8 66BA008E                     mov    dx, 8E00h
000005AC 6689C3                       mov    bx, ax
000005AF 89D8                         mov    eax, ebx     ; /* selector = 0x0008 = cs */
                                                                  ; /* interrupt gate - dpl=0, present */
000005B1 AB                           stosd  ; selector & offset bits 0-15
000005B2 89D0                         mov    eax, edx
```

```
817 000005B4 AB                         stosd  ; attributes & offset bits 16-23
818 000005B5 E2EE                       loop   rp_sidt1
819                                     ; 15/04/2016
820                                     ; TRDOS 386 (TRDOS v2.0) /// 32 sofware interrupts ///
821                                     ;mov   cl, 16        ; 16 software interrupts (INT 30h to INT 3Fh)
822 000005B7 B120                       mov    cl, 32       ; 32 software interrupts (INT 30h to INT 4Fh)
823                                   rp_sidt2:
824 000005B9 AD                         lodsd
825 000005BA 21C0                       and    eax, eax
826 000005BC 7413                       jz     short rp_sidt3
827 000005BE 89C2                       mov    edx, eax
828 000005C0 66BA00EE                   mov    dx, 0EE00h    ; P=1b/DPL=11b/01110b
829 000005C4 6689C3                     mov    bx, ax
830 000005C7 89D8                       mov    eax, ebx      ; selector & offset bits 0-15
831 000005C9 AB                         stosd
832 000005CA 89D0                       mov    eax, edx
833 000005CC AB                         stosd
834 000005CD E2EA                       loop   rp_sidt2
835 000005CF EB16                       jmp    short sidt_OK
836                                   rp_sidt3:
837 000005D1 B8[750D0000]               mov    eax, ignore_int
838 000005D6 89C2                       mov    edx, eax
839 000005D8 66BA00EE                   mov    dx, 0EE00h    ; P=1b/DPL=11b/01110b
840 000005DC 6689C3                     mov    bx, ax
841 000005DF 89D8                       mov    eax, ebx      ; selector & offset bits 0-15
842                                   rp_sidt4:
843 000005E1 AB                         stosd
844 000005E2 92                         xchg   eax, edx
845 000005E3 AB                         stosd
846 000005E4 92                         xchg   edx, eax
847 000005E5 E2FA                       loop   rp_sidt4
848                                   sidt_OK:
849 000005E7 0F011D[5E6D0000]           lidt   [idtd]
850                                     ;
851                                     ; TSS descriptor setup ; 24/03/2015
852 000005EE B8[28810100]               mov    eax, task_state_segment
853 000005F3 66A3[0A6D0000]             mov    [gdt_tss0], ax
854 000005F9 C1C010                     rol    eax, 16
855 000005FC A2[0C6D0000]               mov    [gdt_tss1], al
856 00000601 8825[0F6D0000]             mov    [gdt_tss2], ah
857 00000607 66C705[8E810100]68-        mov    word [tss.IOPB], tss_end - task_state_segment
857 0000060F 00
858                                     ;
859                                     ; IO Map Base address (When this address points
860                                     ; to end of the TSS, CPU does not use IO port
861                                     ; permission bit map for RING 3 IO permissions,
862                                     ; access to any IO ports in ring 3 will be forbidden.)
863                                     ;
864                                     ;mov   [tss.esp0], esp ; TSS offset 4
865                                     ;mov   word [tss.ss0], KDATA ; TSS offset 8 (SS)
866 00000610 66B82800                   mov    ax, TSS  ; It is needed when an interrupt
867                                                     ; occurs (or a system call -software INT- is requested)
868                                                     ; while cpu running in ring 3 (in user mode).
869                                                     ; (Kernel stack pointer and segment will be loaded
870                                                     ; from offset 4 and 8 of the TSS, by the CPU.)
871 00000614 0F00D8                       ltr    ax  ; Load task register
872                                     ;
873                                   esp0_set0:
874                                     ; 30/07/2015
875 00000617 8B0D[94810100]             mov    ecx, [memory_size] ; memory size in pages
876 0000061D C1E10C                     shl    ecx, 12 ; convert page count to byte count
877 00000620 81F900004000               cmp    ecx, CORE ; beginning of user's memory space (400000h)
878                                                      ; (kernel mode virtual address)
879 00000626 7605                       jna    short esp0_set1
880                                     ;
881                                     ; If available memory > CORE (end of the 1st 4 MB)
882                                     ; set stack pointer to CORE
883                                     ;(Because, PDE 0 is reserved for kernel space in user's page directory)
884                                     ;(PDE 0 points to page table of the 1st 4 MB virtual address space)
885 00000628 B900004000                 mov    ecx, CORE
886                                   esp0_set1:
887 0000062D 89CC                       mov    esp, ecx ; top of kernel stack (**tss.esp0**)
888                                   esp0_set_ok:
889                                     ; 30/07/2015 (**tss.esp0**)
890 0000062F 8925[2C810100]             mov    [tss.esp0], esp
891 00000635 66C705[30810100]10-          mov     word [tss.ss0], KDATA
891 0000063D 00
892                                     ; 14/08/2015
893                                     ; 10/11/2014 (Retro UNIX 386 v1 - Erdogan Tan)
894                                     ;
895                                     ;cli   ; Disable interrupts (for CPU)
896                                     ;    (CPU will not handle hardware interrupts, except NMI!)
897                                     ;
898 0000063E 30C0                       xor    al, al       ; Enable all hardware interrupts!
899 00000640 E621                       out    21h, al             ; (IBM PC-AT compatibility)
900 00000642 EB00                       jmp    $+2          ; (All conventional PC-AT hardware
901 00000644 E6A1                       out    0A1h, al     ;  interrupts will be in use.)
902                                                         ; (Even if related hardware component
903                                                         ;  does not exist!)
904                                     ; Enable NMI
905 00000646 B07F                       mov    al, 7Fh             ; Clear bit 7 to enable NMI (again)
906 00000648 E670                       out    70h, al
907                                     ; 23/02/2015
908 0000064A 90                         nop
909 0000064B E471                       in     al, 71h             ; read in 71h just after writing out to 70h
910                                                         ; for preventing unknown state (!?)
911                                     ;
912                                     ; Only a NMI can occur here... (Before a 'STI' instruction)
913                                     ;
914                                     ; 02/09/2014
915 0000064D 6631DB                     xor    bx, bx
916 00000650 66BA0002                   mov    dx, 0200h    ; Row 2, column 0  ; 07/03/2015
917 00000654 E83E1D0000                 call   _set_cpos    ; 24/01/2016
918
919                                     ; 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[54090000]02               cmp    byte [vbe3], 2 ; 02h
00000660 770B                           ja     short vbe3_pmid_chk
                                        ;jb    short display_mem_info
00000662 0F82C2010000                   jb     display_mem_info ; 02/12/2020
00000668 E9F6010000                     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[54090000]                 dec    byte [vbe3] ; 2 = VBE2 compatible
                                                           ; (vbe3 feature is defective in this vbios)
                                        ;jmp   short display_mem_info
                                        ; 02/12/2020
00000688 E99D010000                     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

                                    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
0000069E 89F7                           mov    edi, esi

                                        ;mov   esi, [VBE3INFOBLOCK+22] ; 097E00h + 16h
                                        ;            ; OemVendorNamePtr (seg16:off16)
000006A0 8B35067E0900                   mov    esi, [VBE3INFOBLOCK+6] ; 097E00h + 06h
                                                    ; OemStringPtr (seg16:off16)
000006A6 30C0                           xor    al, al  ; eax = 0
000006A8 6696                           xchg   ax, si ; ax = offset, si = 0
000006AA C1EE0C                         shr    esi, 12 ; (to convert segment to base addr)
000006AD 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
000006B0 B40C                       mov    ah, 0Ch ; Black background, light red forecolor

000006B2 E8D73B0000                 call   print_kmsg

                                    ;mov   ah, 07h

000006B7 BE[397E0100]               mov    esi, vesa_vbe3_bios_msg
                                    ;call  print_kmsg
000006BC E8D33B0000                 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
```

```
1025 0000000E <res 00000002>          .B8000Sel:  resw 1 ; Selector to access B8000h physical mem
1026 00000010 <res 00000002>          .CodeSegSel:      resw 1 ; Selector to access code segment as data
1027 00000012 <res 00000001>          .InProtectMode: resb 1 ; Set to 1 when in protected mode
1028 00000013 <res 00000001>          .Checksum:  resb 1 ; Checksum byte for structure
1029                                  .size:
1030
1031                                  endstruc
1032
1033                                        ; 29/11/2020
1034                                  vbe3pminit:
1035                                        ; 30/11/2020
1036                                        ;cmp   byte [vbe3], 3 ; is VESA VBE3 PMI ready ?
1037                                        ;jne   short di4
1038
1039                                        ; Allocate 64KB contiguous (kernel) memory block
1040 000006C1 31C0                        xor    eax, eax
1041 000006C3 B900000100                  mov    ecx, 65536
1042 000006C8 E8A65D0000                  call   allocate_memory_block
1043                                       ;jc    short di4
1044 000006CD 0F8250010000                jc     di0 ; 30/11/2020
1045
1046                                       ; of course this block must be in the 1st 16MB
1047                                       ; because vbe3 pmi segments will be 16 bit segments
1048                                       ; (80286 type segment descriptors in GDT)
1049
1050 000006D3 A3[20120300]                 mov    [vbe3bios_addr], eax
1051
1052                                       ; set [pmid_addr] to the new location
1053 000006D8 BE00000C00                  mov    esi, 0C0000h
1054
1055                                       ; 30/11/2020
1056 000006DD 29F7                        sub    edi, esi ; izolate offset
1057 000006DF 01C7                        add    edi, eax ; new address
1058 000006E1 893D[24120300]              mov    [pmid_addr], edi ; new 'PMID' location
1059
1060                                       ; Move VIDEO BIOS from 0C0000h to EAX
1061 000006E7 B900400000                  mov    ecx, 65536/4
1062 000006EC 89C7                        mov    edi, eax ; 30/11/2020
1063 000006EE F3A5                        rep    movsd
1064
1065                                       ; 02/12/2020
1066                                       ; 30/11/2020
1067                                       ; set vbe3 segment selectors
1068
1069                                       ; VBE3CS (VESA VBE3 video bios code segment)
1070 000006F0 BF[126D0000]                mov    edi, _vbe3_CS+2 ; base address bits 0..15
1071 000006F5 66AB                        stosw  ; edi = _vbe3_CS+4
1072 000006F7 C1C810                      ror    eax, 16
1073 000006FA 8807                        mov    [edi], al ; base address, bits 16..23
1074
1075                                       ; VBE3DS ('CodeSegSel' in PMInfoBlock)
1076 000006FC BF[3C6D0000]                mov    edi, _vbe3_DS+4 ; base addr bits 16..23
1077 00000701 8807                        mov    [edi], al
1078 00000703 C1C010                      rol    eax, 16
1079 00000706 668947FE                    mov    [edi-2], ax ; base address, bits 0..15
1080
1081                                       ; VBE3BDS (BIOSDataSel in PMInfoBlock)
1082 0000070A BF[1A6D0000]                mov    edi, _vbe3_BDS+2 ; base addr bits 0..15
1083 0000070F B800700900                  mov    eax, VBE3BIOSDATABLOCK ; 1536 bytes
1084 00000714 66AB                        stosw  ; edi = _vbe3_BDS+4
1085 00000716 C1E810                      shr    eax, 16
1086 00000719 8807                        mov    [edi], al ; base address, bits 16..23
1087
1088                                       ; VBE3SS (1024 bytes)
1089 0000071B BF[426D0000]                mov    edi, _vbe3_SS+2 ; base addr bits 0..15
1090 00000720 B800600900                  mov    eax, VBE3STACKADDR ; size = 1024 bytes
1091 00000725 66AB                        stosw  ; edi = _vbe3_SS+4
1092 00000727 C1E810                      shr    eax, 16
1093 0000072A 8807                        mov    [edi], al ; base address, bits 16..23
1094
1095                                       ; stack pointer (esp) will be set to 1020
1096                                       ; (before VBE3 PMI call)
1097
1098                                       ; VBE3ES (max: 2048 bytes)
1099 0000072C BF[4A6D0000]                mov    edi, _vbe3_ES+2 ; base addr bits 0..15
1100 00000731 B800760900                  mov    eax, VBE3SAVERESTOREBLOCK
1101 00000736 66AB                        stosw  ; edi = _vbe3_ES+4
1102 00000738 C1E810                      shr    eax, 16
1103 0000073B 8807                        mov    [edi], al ; base address, bits 16..23
1104
1105                                       ;Note: low word of _VBE3_ES base address will be
1106                                       ;      set -again- by VBE3 PMI caller routine
1107
1108                                       ; 09/12/2020
1109                                       ;; set pmi32 (as VBE3 PMI is ready)
1110                                       ;inc   byte [pmi32] ; = 1
1111
1112                                       ; KCODE16 (set PMI far return segment)
1113 0000073D BF[526D0000]                mov    edi, _16bit_CS+2 ; base addr bits 0..15
1114 00000742 B8[CA070000]                mov    eax, pminit_return_addr16
1115 00000747 66AB                        stosw  ; edi = _16bit_CS+4
1116 00000749 C1E810                      shr    eax, 16
1117 0000074C 8807                        mov    [edi], al ; base address, bits 16..23
1118
1119                                       ; 30/11/2020
1120                                       ; clear mem from VBE3 BIOS data area emu block
1121                                       ; to end of vbe3 buffers
1122
1123                                       ; 01/12/2020
1124 0000074E BF00700900                  mov    edi, VBE3BIOSDATABLOCK ; 97000h
1125                                       ;mov   cx, (VBE3INFOBLOCK-VBE3BIOSDATABLOCK)/4
1126                                       ;      ; ecx = 3584/4 double words
1127                                       ; 21/12/2020
1128 00000753 66B90003                    mov    cx, (VBE3MODEINFOBLOCK-VBE3BIOSDATABLOCK)/4
1129                                             ; ecx = 3072/4 double words
```

```
1130                                    ;xor   eax, eax
1131 00000757 30C0                     xor    al, al
1132 00000759 F3AB                     rep    stosd
1133
1134                                    ; Filling PMInfoBlock selector fields
1135 0000075B 8B3D[24120300]           mov    edi, [pmid_addr]
1136 00000761 66C747083800             mov    word [edi+PMInfo.BIOSDataSel], VBE3BDS
1137 00000767 66C7470A4000             mov    word [edi+PMInfo.A0000Sel], VBE3A000
1138 0000076D 66C7470C4800             mov    word [edi+PMInfo.B0000Sel], VBE3B000
1139 00000773 66C7470E5000             mov    word [edi+PMInfo.B8000Sel], VBE3B800
1140 00000779 66C747105800             mov    word [edi+PMInfo.CodeSegSel], VBE3DS
1141 0000077F C6471201                 mov    byte [edi+PMInfo.InProtectMode], 1
1142
1143                                    ; Calculate and write checksum byte
1144 00000783 89FE                     mov    esi, edi
1145 00000785 B113                     mov    cl, PMInfo.size - 1
1146                                    ;xor   ah, ah
1147                                pmid_chksum:
1148 00000787 AC                       lodsb
1149 00000788 00C4                     add    ah, al
1150 0000078A E2FB                     loop   pmid_chksum
1151 0000078C F6DC                     neg    ah ; 1 -> 255, 255 -> 1
1152 0000078E 8826                     mov    byte [esi], ah  ; checksum
1153
1154                                    ; far call PM initialization
1155                                    ; (VBE3 video bios will return via 'retf')
1156
1157 00000790 668B4706                 mov    ax, [edi+PMInfo.PMInitialize]
1158                                    ; 30/11/2020
1159 00000794 C1E010                   shl    eax, 16 ; save entry address in hw
1160                                    ; ax = 0
1161
1162                                    ; 02/12/2020
1163 00000797 68[EE070000]             push   pminit_ok ; normal, near return address
1164
1165                                    ; 30/11/2020
1166                                _VBE3PMI_fcall:
1167                                    ; ax = function, hw of eax = entry address
1168 0000079C 9C                       pushf  ; save 32 bit flags
1169 0000079D 56                       push   esi ; *
1170 0000079E 55                       push   ebp ; **
1171
1172 0000079F 89E5                     mov    ebp, esp ; save 32 bit stack pointer
1173
1174 000007A1 89C6                     mov    esi, eax
1175
1176 000007A3 FA                       cli
1177
1178                                    ; Disable interrupts (clear interrupt flag)
1179                                    ; Reset Interrupt MASK Registers (Master&Slave)
1180 000007A4 B0FF                     mov    al, 0FFh     ; mask off all interrupts
1181 000007A6 E621                     out    21h, al            ; on master PIC (8259)
1182 000007A8 EB00                     jmp    $+2  ; (delay)
1183 000007AA E6A1                     out    0A1h, al     ; on slave PIC (8259)
1184
1185                                    ; 02/12/2020
1186 000007AC 66B86000                 mov    ax, VBE3SS
1187 000007B0 8ED0                     mov    ss, ax
1188
1189 000007B2 BCFC030000               mov    esp, 1020 ; 30/11/2020
1190
1191                                    ; 01/12/2020
1192                                    ;lss   esp, [stack16]
1193
1194 000007B7 C1E810                   shr    eax, 16      ; now, entry address is in lw
1195
1196                                    ; 30/11/2020 - 16 bit pm selector test (OK)
1197                                    ; (32 bit stack push/pop & retf with 32 bit code segment)
1198                                    ; (16 bit stack push/pop with 16 bit code segment)
1199
1200                                    ; return
1201                                    ;push  KCODE16
1202                                    ;push  0 ; 30/11/2020 (pminit_return_addr16)
1203
1204                                    ; 30/11/2020 (16 bit stack during retf from video bios)
1205 000007BA C7042400007000            mov    dword [esp], KCODE16 << 16
1206                                                        ; ip = 0, cs = KCODE16
1207                                    ; 01/12/2020
1208                                    ;mov   dword [VBE3STACKADDR+1020], KCODE16*65536
1209
1210                                    ;mov   [jumpfar16], eax
1211
1212                                    ; 02/12/2020
1213                                    ; 30/11/2020 (32 bit stack during retf from kernel)
1214                                    ; far jump/call via retf
1215 000007C1 6A30                     push   VBE3CS ; VBE3 video bios's code segment
1216 000007C3 50                       push   eax ; PMInitialize or EntryPoint
1217
1218 000007C4 6689F0                   mov    ax, si ; restore function
1219
1220                                    ; 02/12/2020
1221 000007C7 31F6                     xor    esi, esi ; (not necessary, it is not used)
1222
1223 000007C9 CB                       retf   ; far return (to 16 bit code segment)
1224
1225                                    ; 01/12/2020
1226                                    ;db    0EAh  ; far jump to 16 bit code segment
1227                                ;jumpfar16:
1228                                    ;dd    0
1229                                    ;dw    VBE3CS
1230
1231                                ;stack16:
1232                                    ;dd    1020
1233                                    ;dw    VBE3SS
1234
```

```
1235                                    align 2
1236
1237                                pminit_return_addr16:
1238                                    ; 02/12/2020
1239                                    ; 30/11/2020
1240                                    ;;db   66h              ; Prefix for 32-bit
1241                                    ;db    0EAh             ; Opcode for far jump
1242                                    ;dd    pminit_return_addr32 ; 32 bit Offset
1243                                    ;dw    KCODE            ; 32 bit code segment
1244                                    ; 01/12/2020
1245 000007CA EA[D1070000]0800         jmp    KCODE:pminit_return_addr32
1246
1247                                pminit_return_addr32:
1248                                    ; restore 32 bit kernel selectors and 32 bit stack addr
1249 000007D1 BE10000000               mov    esi, KDATA
1250 000007D6 8EDE                     mov    ds, si
1251 000007D8 8EC6                     mov    es, si
1252 000007DA 8ED6                     mov    ss, si
1253 000007DC 89EC                     mov    esp, ebp  ; top of stack = iretd return addr
1254
1255 000007DE 5D                       pop    ebp ; **
1256 000007DF 5E                       pop    esi ; *
1257 000007E0 9D                       popf   ; restore 32 bit flags
1258
1259                                    ; enable interrupts
1260
1261 000007E1 FA                       cli
1262
1263                                    ; 21/12/2020
1264 000007E2 50                       push   eax
1265
1266 000007E3 30C0                     xor    al, al    ; Enable all hardware interrupts!
1267 000007E5 E621                     out    21h, al         ; (IBM PC-AT compatibility)
1268 000007E7 EB00                     jmp    $+2       ; (All conventional PC-AT hardware
1269 000007E9 E6A1                     out    0A1h, al   ;  interrupts will be in use.)
1270                                                     ; (Even if related hardware component
1271                                                     ;  does not exist!)
1272 000007EB 58                       pop    eax
1273
1274 000007EC FB                       sti
1275
1276                                    ; top of stack = return address
1277                                    ; ('pminit_ok' for PMinit)
1278
1279 000007ED C3                       retn
1280
1281                                pminit_ok:
1282                                    ; 03/12/2020
1283                                    ; (set [pmid_addr] to PMI entry point for next calls)
1284 000007EE 8305[24120300]04         add    dword [pmid_addr], PMInfo.EntryPoint ; + 4
1285
1286                                    ; 17/01/2021
1287                                    ; copy EDID data from temporary location to final address
1288 000007F5 803D[25430000]4F         cmp    byte [edid], 4Fh
1289 000007FC 7511                     jne    short vbe3h_chcl
1290 000007FE B920000000               mov    ecx, 32 ; 128 bytes, 32 dwords
1291 00000803 BE007D0900               mov    esi, VBE3EDIDINFOBLOCK ; 97D00h
1292 00000808 BF[9C110300]             mov    edi, edid_info
1293 0000080D F3A5                     rep    movsd
1294                                    ; 17/01/2021
1295                                vbe3h_chcl:
1296                                    ; 16/01/2021
1297                                    ;; 06/12/2020
1298                                    ;; Save video mode 03h regs/dac/bios state
1299                                    ;
1300                                    ;mov   ax, 4F04h ; VESA VBE Function 04h
1301                                    ;              ; Save/Restore State
1302                                    ;sub   dl, dl ; 0 = return buffer size
1303                                    ;mov   cx, 0Fh  ; ctrl/bios/dac/regs
1304                                    ;
1305                                    ;call  int10h_32bit_pmi
1306                                    ;; bx = number of 64-byte blocks to hold the state buff
1307                                    ;
1308                                    ;;mov  [vbe3stbufsize], bx
1309                                    ;; 16/01/2021
1310                                    ;or    word [vbe3stbsflags], 32768  ; set bit 15
1311                                    ;mov   ax, bx
1312                                    ;shl   ax, 6  ; * 64
1313                                    ;mov   [vbestatebufsize+30], ax
1314                                    ;
1315                                    ;; 06/12/2020
1316                                    ;; check 'vbe3stbufsize' (it must be <= 32)
1317                                    ;
1318                                    ;cmp   bx, 32
1319                                    ;ja    short display_mem_info ; light red forecolor
1320                                    ;
1321                                    ;; 16/01/2021
1322                                    ;or    byte [vbe3stbsflags], 1 ; set bit 0
1323
1324                                    ; 30/11/2020
1325                                    ; Change VESA VBE3 BIOS text color in order to give
1326                                    ; "VBE3 PMI initialization has been successed" meaning
1327
1328 0000080F BE40810B00                mov    esi, 0B8000h + 160*2 ; row 2
1329 00000814 89F7                     mov    edi, esi
1330                                vbe3h_chcl_next:
1331 00000816 66AD                     lodsw
1332 00000818 80FC0C                   cmp    ah, 0Ch      ; light red forecolor
1333 0000081B 750D                     jne    short display_mem_info
1334 0000081D B40E                     mov    ah, 0Eh ; yellow forecolor
1335 0000081F 66AB                     stosw
1336 00000821 EBF3                     jmp    short vbe3h_chcl_next
1337
1338                                di0:
1339                                    ; 30/11/2020
```

```
                                        ; Memory allocation error !
00000823 C605[54090000]00               mov   byte [vbe3], 0 ; disable VBE3

                                    display_mem_info:
                                        ; 19/12/2020
                                        ; temporary
0000082A E87B3A0000                     call  default_lfb_info
                                        ;
                                        ; 06/11/2014
0000082F E8013A0000                     call  memory_info
                                        ; 14/08/2015
                                        ;call getch ; 28/02/2015
                                    drv_init:
00000834 FB                             sti   ; Enable Interrupts
                                        ; 06/02/2015
00000835 8B15[F06D0000]                 mov   edx, [hd0_type] ; hd0, hd1, hd2, hd3
0000083B 668B1D[EE6D0000]               mov   bx, [fd0_type] ; fd0, fd1
                                        ; 22/02/2015
00000842 6621DB                         and   bx, bx
00000845 756C                           jnz   short di1
                                        ;
00000847 09D2                           or    edx, edx
00000849 757A                           jnz   short di2
                                        ;
                                    setup_error:
0000084B BE[A6440100]                   mov   esi, setup_error_msg
                                    psem:
00000850 AC                             lodsb
00000851 08C0                           or    al, al
                                        ;jz   short haltx ; 22/02/2015
00000853 747B                           jz    short di3
00000855 56                             push  esi
                                        ; 13/05/2016
00000856 BB07000000                     mov   ebx, 7 ; Black background,
                                              ; light gray forecolor
                                              ; Video page 0 (BH=0)
0000085B E89D1A0000                     call  _write_tty
00000860 5E                             pop   esi
00000861 EBED                           jmp   short psem

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

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

00000863 29C0                           sub   eax, eax ; 0
00000865 66BACE01                       mov   dx, 1CEh ; VBE_DISPI_IOPORT_INDEX
00000869 66EF                           out   dx, ax ; VBE_DISPI_INDEX_ID register
                                        ;mov  ax, 0B0C0h ; VBE_DISPI_ID0
                                        ;mov  dx, 1CFh ; VBE_DISPI_IOPORT_DATA
0000086B 6642                           inc   dx
                                        ;out  dx, ax
                                        ;nop
0000086D 66ED                           in    ax, dx
0000086F 80FCB0                         cmp   ah, 0B0h
                                        ;jne  short not_boch_qemu_vbe
00000872 75B6                           jne   short display_mem_info
00000874 3CC5                           cmp   al, 0C5h ; it must be 0B0C4h or 0B0C5h ..
                                        ;ja   short not_boch_qemu_vbe
00000876 77B2                           ja    short display_mem_info
00000878 3CC0                           cmp   al, 0C0h ; 0B0C0h to 0B0C5h .. ; Qemu
                                        ;cmp  al, 0C4h ; 0BC04h or 0B0C5h is OK ; Bochs
                                        ;jb   short not_boch_qemu_vbe
0000087A 72AE                           jb    short display_mem_info

                                        ; save VESA VBE2 bios (bochs/qemu) signature
                                        ; for enabling VBE2 functions in TRDOS 386 v2 kernel
0000087C A2[55090000]                   mov   [vbe2bios], al ; 0C4h or 0C5h (for BOCHS)
                                                             ; (0C0h-0C5h for QEMU)
00000881 C605[427E0100]32               mov   byte [vbe_vnumber], "2"
                                        ; 26/11/2020
                                        ; "BOCHS/QEMU/VIRTUALBOX VBE2 Video BIOS ..".
00000888 BE[247E0100]                   mov   esi, vbe2_bochs_vbios ; BOCH/QEMU vbios msg
0000088D B40E                           mov   ah, 0Eh  ; Yellow font
0000088F E8FA390000                     call  print_kmsg

                                        ; this is not necessary ! (20/11/2020)
00000894 803D[55090000]C4               cmp   byte [vbe2bios], 0C4h
0000089B 728D                           jb    display_mem_info   ; (QEMU)

                                        ; Display kernel version message if 0E9h hack port
                                        ; is enabled (bochs emulator feature)
0000089D 66BAE900                       mov   dx, 0E9h ; hack port for BOCHS
000008A1 BE[7D7E0100]                   mov   esi, kernel_version_msg
                                    kvmsg_next_char:
000008A6 AC                             lodsb
000008A7 08C0                           or    al, al
000008A9 7505                           jnz   short put_kvmsg_in_hack_port
                                    not_boch_qemu_vbe:
000008AB E97AFFFFFF                     jmp   display_mem_info
                                    put_kvmsg_in_hack_port:
000008B0 EE                             out   dx, al
000008B1 EBF3                           jmp   short kvmsg_next_char

                                    di1:
                                        ; supress 'jmp short T6'
                                        ;  (activate fdc motor control code)
000008B3 66C705[B6090000]90-            mov   word [T5], 9090h ; nop
```

```
1444 000008BB 90
1445                                  ;
1446                                  ;mov   ax, int_0Eh  ; IRQ 6 handler
1447                                  ;mov   di, 0Eh*4    ; IRQ 6 vector
1448                                  ;stosw
1449                                  ;mov   ax, cs
1450                                  ;stosw
1451                                  ;; 16/02/2015
1452                                    ;;mov     dword [DISKETTE_INT], fdc_int ; IRQ 6 handler
1453                                  ;
1454 000008BC E820490000              CALL   DSKETTE_SETUP; Initialize Floppy Disks
1455                                  ;
1456 000008C1 09D2                    or     edx, edx
1457 000008C3 740B                      jz      short di3
1458                                di2:
1459 000008C5 E85E490000              call         DISK_SETUP   ; Initialize Fixed Disks
1460 000008CA 0F827BFFFFFF              jc      setup_error
1461                                di3:
1462 000008D0 E834390000              call   setup_rtc_int; 22/05/2015 (dsectrpm.s)
1463                                  ;
1464 000008D5 E8623A0100              call   display_disks ; 07/03/2015  (Temporary)
1465                                ;haltx:
1466                                  ; 14/08/2015
1467                                  ;call  getch ; 22/02/2015
1468                                  ;sti   ; Enable interrupts (for CPU)
1469                                ;     ; 29/01/2016
1470                                ;     sub    ah, ah ;  read time count
1471                                ;     call   int1Ah
1472                                ;     mov    edx, ecx ; 18.2 * seconds
1473                                ;md_info_msg_wait1:
1474                                ;     ; 29/01/2016
1475                                ;     mov    ah, 1
1476                                ;     call   int16h
1477                                ;     jz     short md_info_msg_wait2
1478                                ;     xor    ah, ah ; 0
1479                                ;       call    int16h
1480                                ;     jmp    short md_info_msg_ok
1481                                ;md_info_msg_wait2:
1482                                ;     sub    ah, ah  ; read time count
1483                                ;     call   int1Ah
1484                                ;     cmp    edx, ecx ; ; 18.2 * seconds
1485                                ;     jna    short md_info_msg_wait3
1486                                ;     xchg   edx, ecx
1487                                ;md_info_msg_wait3:
1488                                ;     sub    ecx, edx
1489                                ;     cmp    ecx, 127 ; 7 seconds (18.2 * 7)
1490                                ;     jb     short md_info_msg_wait1
1491                                ;md_info_msg_ok:
1492
1493                                  ; 15/12/2020
1494                                  ; set initial values of LFB parameters
1495
1496 000008DA 803D[54090000]02          cmp    byte [vbe3], 2
1497 000008E1 7214                    jb     short di4
1498
1499                                  ;mov   ax, [def_LFB_addr]
1500                                  ;shl   eax, 16
1501                                  ;mov   [LFB_ADDR], eax
1502                                  ;mov   eax, 1024*768*3
1503                                  ;mov   [LFB_SIZE], eax
1504
1505 000008E3 BEFE7B0900                mov    esi, VBE3MODEINFOBLOCK - 2
1506 000008E8 66C7061801              mov    word [esi], 0118h ; default vbe mode
1507                                                            ; 1024*768, 24bpp
1508 000008ED E8A3310000              call   set_lfbinfo_table
1509
1510 000008F2 E8D2600000              call   allocate_lfb_pages_for_kernel
1511                                di4:
1512                                  ; 08/09/2016
1513 000008F7 0F20C0                  mov    eax, cr0
1514 000008FA A810                    test   al, 10h  ; Bit 4, ET (Extension Type)
1515 000008FC 7408                    jz     short sysinit
1516                                  ; 27/02/2017
1517 000008FE FE05[4C8E0100]          inc    byte [fpready]
1518                                  ; 80387 (FPU) is ready
1519 00000904 DBE3                    fninit ; Initialize Floating-Point Unit
1520                                sysinit:
1521                                  ; 30/06/2015
1522 00000906 E81B6B0000              call   sys_init
1523                                  ;
1524                                  ;jmp   cpu_reset ; 22/02/2015
1525                                hang:
1526                                  ; 23/02/2015
1527                                  ;sti                ; Enable interrupts
1528 0000090B F4                      hlt
1529                                  ;
1530                                  ;nop
1531                                  ;; 03/12/2014
1532                                  ;; 28/08/2014
1533                                  ;mov   ah, 11h
1534                                  ;call  getc
1535                                  ;jz      _c8
1536                                  ;
1537                                  ; 23/02/2015
1538                                  ; 06/02/2015
1539                                  ; 07/09/2014
1540 0000090C 31DB                    xor    ebx, ebx
1541 0000090E 8A1D[BE810100]          mov    bl, [ptty]   ; active_page
1542 00000914 89DE                    mov    esi, ebx
1543 00000916 66D1E6                  shl    si, 1
1544 00000919 81C6[C0810100]          add    esi, ttychr
1545 0000091F 668B06                  mov    ax, [esi]
1546 00000922 6621C0                  and    ax, ax
1547                                  ;jz    short _c8
1548 00000925 74E4                    jz     short hang
```

```
1549 00000927 66C7060000                mov    word [esi], 0
1550 0000092C 80FB03                    cmp    bl, 3        ; Video page 3
1551                                    ;jb    short _c8
1552 0000092F 72DA                      jb     short hang
1553                                    ;
1554                                    ; 13/05/2016
1555                                    ; 07/09/2014
1556                            nxtl:
1557 00000931 6653                      push   bx
1558 00000933 66BB0E00                  mov    bx, 0Eh      ; Yellow character
1559                                                        ; on black background
1560                                                        ; bh = 0 (video page 0)
1561                                                        ; Retro UNIX 386 v1 - Video Mode 0
1562                                                        ; (PC/AT Video Mode 3 - 80x25 Alpha.)
1563 00000937 6650                      push   ax
1564 00000939 E8BF190000                call   _write_tty
1565 0000093E 6658                      pop    ax
1566 00000940 665B                      pop    bx
1567 00000942 3C0D                      cmp    al, 0Dh             ; carriage return (enter)
1568                                    ;jne   short _c8
1569 00000944 75C5                      jne    short hang
1570 00000946 B00A                      mov    al, 0Ah             ; next line
1571 00000948 EBE7                      jmp    short nxtl
1572
1573                            ;_c8:
1574                            ;       ; 25/08/2014
1575                            ;       cli                        ; Disable interrupts
1576                            ;       mov    al, [scounter + 1]
1577                            ;       and    al, al
1578                            ;       jnz    hang
1579                            ;       call   rtc_p
1580                            ;       jmp     hang
1581
1582                                    ; 27/08/2014
1583                                    ; 20/08/2014
1584                            printk:
1585                                      ;mov    edi, [scr_row]
1586                            pkl:
1587 0000094A AC                        lodsb
1588 0000094B 08C0                      or     al, al
1589 0000094D 7404                      jz     short pkr
1590 0000094F 66AB                      stosw
1591 00000951 EBF7                      jmp    short pkl
1592                            pkr:
1593 00000953 C3                        retn
1594
1595                            ; 14/11/2020 (TRDOS 386 v2.0.3)
1596 00000954 00                vbe3: db 0  ; VESA VBE version (must be 03h)
1597                                        ; for using video bios calls in protected mode
1598                            vbe2bios:
1599 00000955 B0                        db 0B0h ;
1600                            ;pmid_addr:
1601                                    ;dw 0  ; > 0 if 'PMID' sign is found
1602                                    ;       ;       ('pmid' offset addr in VGA bios seg, 0C000h)
1603                                    ;; 02/12/2020
1604                                    ;dw 0  ; 32 bit address in pmid_addr
1605
1606                            ; 28/02/2017
1607                            ; 22/01/2017
1608                            ; 15/01/2017
1609                            ; 14/01/2017
1610                            ; 02/01/2017
1611                            ; 25/12/2016
1612                            ; 19/12/2016
1613                            ; 10/12/2016 (callback)
1614                            ; 06/06/2016
1615                            ; 23/05/2016
1616                            ; 22/05/2016 - TRDOS 386 (TRDOS v2.0) Timer Event Modifications
1617                            ; 25/07/2015
1618                            ; 14/05/2015 (multi tasking -time sharing- 'clock', x_timer)
1619                            ; 17/02/2015
1620                            ; 06/02/2015 (unix386.s)
1621                            ; 11/12/2014 - 22/12/2014 (dsectrm2.s)
1622                            ;
1623                            ; IBM PC-XT Model 286 Source Code - BIOS2.ASM (06/10/85)
1624                            ;
1625                            ;-- HARDWARE INT  08 H - ( IRQ LEVEL 0 ) ---------------------------------------
1626                            ;     THIS ROUTINE HANDLES THE TIMER INTERRUPT FROM FROM CHANNEL 0 OF        :
1627                            ;     THE 8254 TIMER.  INPUT FREQUENCY IS 1.19318 MHZ AND THE DIVISOR        :
1628                            ;     IS 65536, RESULTING IN APPROXIMATELY 18.2 INTERRUPTS EVERY SECOND.     :
1629                            ;                                                                            :
1630                            ;     THE INTERRUPT HANDLER MAINTAINS A COUNT (40:6C) OF INTERRUPTS SINCE    :
1631                            ;     POWER ON TIME, WHICH MAY BE USED TO ESTABLISH TIME OF DAY.        :
1632                            ;     THE INTERRUPT HANDLER ALSO DECREMENTS THE MOTOR CONTROL COUNT (40:40)  :
1633                            ;     OF THE DISKETTE, AND WHEN IT EXPIRES, WILL TURN OFF THE           :
1634                            ;     DISKETTE MOTOR(s), AND RESET THE MOTOR RUNNING FLAGS.                   :
1635                            ;     THE INTERRUPT HANDLER WILL ALSO INVOKE A USER ROUTINE THROUGH           :
1636                            ;     INTERRUPT 1CH AT EVERY TIME TICK.  THE USER MUST CODE A           :
1637                            ;     ROUTINE AND PLACE THE CORRECT ADDRESS IN THE VECTOR TABLE.        :
1638                            ;-----------------------------------------------------------------------------
1639                            ;
1640
1641                            timer_int:  ; IRQ 0
1642                            ;int_08h:   ; Timer
1643                                  ; 14/10/2015
1644                                  ; Here, we are simulating system call entry (for task switch)
1645                                  ; (If multitasking is enabled,
1646                                  ; 'clock' procedure may jump to 'sysrelease')
1647
1648 00000956 1E                      push   ds
1649 00000957 06                      push   es
1650 00000958 0FA0                    push   fs
1651 0000095A 0FA8                    push   gs
1652
1653 0000095C 60                      pushad ; eax, ecx, edx, ebx, esp -before pushad-, ebp, esi, edi
```

```
1654 0000095D 66B91000                  mov     cx, KDATA
1655 00000961 8ED9                        mov     ds, cx
1656 00000963 8EC1                        mov     es, cx
1657 00000965 8EE1                        mov     fs, cx
1658 00000967 8EE9                        mov     gs, cx
1659
1660 00000969 0F20D9                    mov    ecx, cr3
1661 0000096C 890D[5C040300]            mov    [cr3reg], ecx ; save current cr3 register value/content
1662
1663                                    ; 14/01/2017
1664 00000972 3B0D[90810100]            cmp    ecx, [k_page_dir]
1665 00000978 7409                      je     short T3
1666
1667 0000097A 8B0D[90810100]            mov    ecx, [k_page_dir]
1668 00000980 0F22D9                    mov    cr3, ecx
1669                                T3:
1670                                    ;sti                        ; INTERRUPTS BACK ON
1671 00000983 66FF05[10820100]          INC    word [TIMER_LOW]     ; INCREMENT TIME
1672 0000098A 7507                      JNZ    short T4             ; GO TO TEST_DAY
1673 0000098C 66FF05[12820100]          INC    word [TIMER_HIGH]    ; INCREMENT HIGH WORD OF TIME
1674                                T4:                             ; TEST_DAY
1675 00000993 66833D[12820100]18        CMP    word [TIMER_HIGH],018H     ; TEST FOR COUNT EQUALING 24 HOURS
1676 0000099B 7519                      JNZ    short T5             ; GO TO DISKETTE_CTL
1677 0000099D 66813D[10820100]B0-       CMP    word [TIMER_LOW],0B0H
1677 000009A5 00
1678 000009A6 750E                      JNZ    short T5             ; GO TO DISKETTE_CTL
1679
1680                                ;-----      TIMER HAS GONE 24 HOURS
1681                                    ;;SUB  AX,AX
1682                                    ;MOV   [TIMER_HIGH],AX
1683                                    ;MOV   [TIMER_LOW],AX
1684 000009A8 29C0                      sub    eax, eax
1685 000009AA A3[10820100]              mov    [TIMER_LH], eax
1686                                    ;
1687 000009AF C605[14820100]01          MOV    byte [TIMER_OFL],1
1688
1689                                ;-----      TEST FOR DISKETTE TIME OUT
1690
1691                                T5:
1692                                    ; 23/12/2014
1693 000009B6 EB1D                      jmp    short T6             ; will be replaced with nop, nop
1694                                                                ; (9090h) if a floppy disk
1695                                                                ; is detected.
1696                                    ;mov   al,[CS:MOTOR_COUNT]
1697 000009B8 A0[17820100]              mov    al, [MOTOR_COUNT]
1698 000009BD FEC8                      dec    al
1699                                    ;mov   [CS:MOTOR_COUNT], al      ; DECREMENT DISKETTE MOTOR CONTROL
1700 000009BF A2[17820100]              mov    [MOTOR_COUNT], al
1701                                    ;mov   [ORG_MOTOR_COUNT], al
1702 000009C4 750F                      JNZ    short T6             ; RETURN IF COUNT NOT OUT
1703 000009C6 B0F0                      mov    al,0F0h
1704                                    ;AND   [CS:MOTOR_STATUS],al      ; TURN OFF MOTOR RUNNING BITS
1705 000009C8 2005[16820100]            and    [MOTOR_STATUS], al
1706                                    ;and   [ORG_MOTOR_STATUS], al
1707 000009CE B00C                      MOV    AL,0CH               ; bit 3 = enable IRQ & DMA,
1708                                                                ; bit 2 = enable controller
1709                                                                ;      1 = normal operation
1710                                                                ;      0 = reset
1711                                                                ; bit 0, 1 = drive select
1712                                                                ; bit 4-7 = motor running bits
1713 000009D0 66BAF203                  MOV    DX,03F2H             ; FDC CTL PORT
1714 000009D4 EE                        OUT    DX,AL                ; TURN OFF THE MOTOR
1715                                T6:
1716                                    ;inc   word [CS:wait_count]      ; 22/12/2014 (byte -> word)
1717                                                                ; TIMER TICK INTERRUPT
1718                                    ;;inc  word [wait_count] ;;27/02/2015
1719                                    ;INT   1CH                  ; TRANSFER CONTROL TO A USER ROUTINE
1720                                    ;cli
1721 000009D5 E857040000                call   u_timer                  ; TRANSFER CONTROL TO A USER ROUTINE
1722                                    ; 23/05/2016
1723 000009DA E89D1A0100                call   clock                ; Multi Tasking control procedure
1724                                T7:
1725                                    ; 14/10/2015
1726 000009DF B020                      MOV    AL,EOI               ; GET END OF INTERRUPT MASK
1727 000009E1 FA                        CLI                         ; DISABLE INTERRUPTS TILL STACK CLEARED
1728 000009E2 E620                      OUT    INTA00,AL            ; END OF INTERRUPT TO 8259 - 1
1729                                    ;
1730                              rtc_int_2:
1731                                    ; 26/12/2016
1732                                    ;mov   ecx, [cr3reg]
1733                                    ; 13/01/2017
1734 000009E4 803D[D4030300]00          cmp    byte [u.t_lock], 0        ; T_LOCK
1735 000009EB 7730                      ja     short timer_int_return  ; Timer Lock : 'sysrele' is needed !
1736                                    ; 28/02/2017
1737                                    ; We need to exit if the user's IRQ callback service is in progress!
1738                                    ; (To prevent a conflict!)
1739 000009ED 803D[D8030300]00          cmp    byte [u.r_lock], 0  ; R_LOCK, IRQ callback service lock !
1740 000009F4 7727                      ja     short timer_int_return  ; Timer Lock : 'sysrele' is needed !
1741                                    ; 15/01/2017
1742 000009F6 803D[208E0100]02          cmp    byte [priority], 2
1743 000009FD 733A                      jnb    short T8  ; current process has a timer event (15/01/2017)
1744                                    ; 22/05/2016
1745 000009FF 803D[218E0100]00          cmp    byte [p_change], 0 ; in 'set_run_sequence', in 'rtc_p'
1746 00000A06 7615                      jna    short timer_int_return ; 23/05/2016
1747
1748                                    ; 15/01/2017
1749
1750                                    ; present process must be changed with high priority process
1751                                    ;xor   al, al
1752 00000A08 31C0                      xor    eax, eax ; 26/12/2016
1753 00000A0A A2[218E0100]              mov    [p_change], al ; 0
1754                                    ;mov   byte [priority], 2 ; 15/01/2017 (there is a timer event)
1755
1756 00000A0F 803D[5B030300]FF          cmp     byte [sysflg], 0FFh ; user or system space ?
1757 00000A16 7416                      je     short rtc_int_3     ; user space ([sysflg]= 0FFh)
```

```
1758
1759                                    ; system space, wait for 'sysret'
1760                                    ; to change running process
1761                                    ; with high priority (event) process
1762
1763 00000A18 A2[A8030300]                      mov    [u.quant], al ; 0
1764
1765                                  timer_int_return: ; 23/05/2016 - jump from 'rtc_int' ('rtc_int_2')
1766 00000A1D 8B0D[5C040300]                    mov    ecx, [cr3reg]       ; previous value/content of cr3 register
1767 00000A23 0F22D9                            mov    cr3, ecx     ; restore cr3 register content
1768                                            ;
1769 00000A26 61                                popad ; edi, esi, ebp, temp (icrement esp by 4), ebx, edx, ecx, eax
1770                                            ;
1771 00000A27 0FA9                              pop    gs
1772 00000A29 0FA1                              pop    fs
1773 00000A2B 07                                pop    es
1774 00000A2C 1F                                pop    ds
1775                                            ;
1776 00000A2D CF                                iretd  ; return from interrupt
1777
1778                                  rtc_int_3:
1779 00000A2E FE05[5B030300]                    inc    byte [sysflg]       ; now, we are in system space
1780                                            ;
1781 00000A34 E9D3CE0000                          jmp     sysrelease ; change running process immediatelly
1782
1783                                  T8:
1784                                            ; 13/01/2017 (eax -> ebx)
1785                                            ; callback checking... (19/12/2016)
1786 00000A39 31DB                              xor    ebx, ebx
1787 00000A3B 871D[D0030300]                    xchg   ebx, [u.tcb] ; callback address (0 = normal return)
1788 00000A41 09DB                              or     ebx, ebx
1789 00000A43 74D8                              jz     short timer_int_return
1790
1791                                            ; Set user's callback routine as return address from this interrupt
1792                                            ; and set normal return address as return address from callback
1793                                            ; routine!!! (19/12/2016)
1794
1795                                            ; 14/01/2017
1796                                            ; 13/01/2017 - Timer Lock (T_LOCK)
1797 00000A45 FE05[D4030300]                    inc    byte [u.t_lock]
1798 00000A4B 8A0D[5B030300]                    mov    cl, [sysflg]
1799 00000A51 880D[D5030300]                    mov    [u.t_mode], cl
1800
1801 00000A57 8B2D[2C810100]                    mov    ebp, [tss.esp0] ; kernel stack address (for ring 0)
1802 00000A5D 83ED14                            sub    ebp, 20             ; eip, cs, eflags, esp, ss
1803 00000A60 892D[5C030300]                    mov    [u.sp], ebp
1804 00000A66 8925[60030300]                    mov    [u.usp], esp
1805
1806                                            ;or    word [ebp+8], 200h ; 22/01/2017, force enabling interrupts
1807
1808 00000A6C 8B44241C                          mov    eax, [esp+28] ; pushed eax
1809 00000A70 A3[64030300]                      mov    [u.r0], eax
1810
1811 00000A75 E8FE060100                        call   wswap ; save user's registers & status
1812
1813                                            ; software int is in ring 0 but timer int must return to ring 3
1814                                            ; so, ring 3 return address and stack registers
1815                                            ; (eip, cs, eflags, esp, ss)
1816                                            ; must be copied to timer int return
1817                                            ; eip will be replaced by callback service routine address
1818
1819 00000A7A C605[5B030300]FF                  mov    byte [sysflg], 0FFh ; user mode
1820
1821                                            ; system mode (system call)
1822                                            ;mov   ebp, [u.sp] ; EIP (u), CS (UCODE), EFLAGS (u),
1823                                                           ; ESP (u), SS (UDATA)
1824
1825 00000A81 8B4510                            mov    eax, [ebp+16]; SS (UDATA
1826 00000A84 89E6                              mov    esi, esp
1827 00000A86 50                                push   eax
1828 00000A87 50                                push   eax
1829 00000A88 89E7                              mov    edi, esp
1830 00000A8A 893D[60030300]                    mov    [u.usp], edi
1831 00000A90 B908000000                        mov    ecx, ((ESPACE/4) - 4) ; except DS, ES, FS, GS
1832 00000A95 F3A5                              rep    movsd
1833 00000A97 B104                              mov    cl, 4
1834 00000A99 F3AB                              rep    stosd
1835 00000A9B 893D[5C030300]                    mov    [u.sp], edi
1836 00000AA1 89EE                              mov    esi, ebp
1837 00000AA3 B105                              mov    cl, 5 ; EIP (u), CS (UCODE), EFLAGS (u), ESP (u), SS (UDATA)
1838 00000AA5 F3A5                              rep    movsd
1839
1840 00000AA7 8B0D[B8030300]                    mov    ecx, [u.pgdir]
1841 00000AAD 890D[5C040300]                    mov    [cr3reg], ecx
1842
1843                                            ; 13/01/207 (eax -> ebx)
1844                                            ; EBX = callback routine address (virtual, not physical address!)
1845
1846                                            ; 09/01/2017
1847                                            ; !!! CALLBACK ROUTINE MUST BE ENDED/RETURNED WITH 'sysrele'
1848                                            ;     system call !!!
1849                                            ; 25/12/2016
1850                                            ; Callback Note: (19/12/2016)
1851                                            ; !!! CALLBACK ROUTINE MUST BE ENDED/RETURNED WITH 'RETN' !!!
1852                                            ;     pushf ; save flags
1853                                            ;     <callback service code>
1854                                            ;     popf  ; restore flags
1855                                            ;     retn ; return to normal running address
1856                                            ;
1857
1858                                            ; 15/01/2017
1859                                            ; 14/01/2017
1860                                            ; 13/01/2017 (eax -> ebx)
1861                                            ; 10/01/2017
1862                                  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
00000AB3 8B2D[5C030300]         mov   ebp, [u.sp]; kernel's stack, points to EIP (user)
00000AB9 895D00                 mov   [ebp], ebx
00000ABC 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
00000AC1 1E                     push  ds ; ** ; 23/05/2016
00000AC2 50                     push  eax ; *
00000AC3 66B81000               mov   ax, KDATA
00000AC7 8ED8                   mov   ds, ax
                                ;
00000AC9 8A25[0E820100]         mov   ah, [RTC_2Hz] ;  2 Hz interrupt to 1 Hz function
00000ACF 80F401                 xor   ah, 1
00000AD2 8825[0E820100]         mov   [RTC_2Hz], ah ; 1 = 0.5 second, 0 = 1 second
00000AD8 753B                   jnz   short rtc_int_return ; half second
                                ; 1 second
                              rtc_int_0:
                                ; 22/05/2016
00000ADA 58                     pop   eax ; *
                                ;
                                ; 14/10/2015 ('timer_int')
                                ; Here, we are simulating system call entry (for task switch)
                                ; (If multitasking is enabled,
                                ; 'clock' procedure may jump to 'sysrelease')
                                ;push  ds ; ** ; 23/05/2016
00000ADB 06                     push  es
00000ADC 0FA0                   push  fs
00000ADE 0FA8                   push  gs
00000AE0 60                     pushad  ; eax, ecx, edx, ebx, esp -before pushad-, ebp, esi, edi
00000AE1 66B91000                 mov     cx, KDATA
                                    ;mov    ds, cx ; 06/06/2016
```

```
1968 00000AE5 8EC1                          mov     es, cx
1969 00000AE7 8EE1                          mov     fs, cx
1970 00000AE9 8EE9                          mov     gs, cx
1971                                        ;
1972 00000AEB 0F20D9                        mov    ecx, cr3
1973 00000AEE 890D[5C040300]                mov    [cr3reg], ecx ; save current cr3 register value/content
1974                                        ;
1975 00000AF4 803D[D4030300]00              cmp    byte [u.t_lock], 0 ; timer lock (callback) status ?
1976 00000AFB 7711                          ja     short rtc_int_1        ; yes
1977
1978                                        ; 15/01/2017
1979 00000AFD 3B0D[90810100]                cmp    ecx, [k_page_dir]
1980 00000B03 7409                          je     short rtc_int_1
1981
1982 00000B05 8B0D[90810100]                mov    ecx, [k_page_dir]
1983 00000B0B 0F22D9                        mov    cr3, ecx
1984                                rtc_int_1:
1985                                        ; Timer event (kernel) functions must be performed with
1986                                        ; 1 second intervals - TRDOS 386 (TRDOS v2.0) feature ! -
1987                                        ;
1988                                        ; 25/08/2014
1989 00000B0E E81A030000                    call   rtc_p  ; 19/05/2016 - major modification
1990
1991                                        ; 23/05/2016
1992 00000B13 28E4                          sub    ah, ah ; 0
1993                                        ; 22/05/2016 - TRDOS 386 timer event modifications
1994                                rtc_int_return: ; 19/05/2016
1995                                        ; 22/02/2015 - dsectpm.s
1996                                        ; [ source: http://wiki.osdev.org/RTC ]
1997                                        ; read status register C to complete procedure
1998                                        ;(it is needed to get a next IRQ 8)
1999 00000B15 B00C                          mov    al, 0Ch ;
2000 00000B17 E670                          out    70h, al ; select register C
2001 00000B19 90                            nop
2002 00000B1A E471                          in     al, 71h ; just throw away contents
2003                                        ; 22/02/2015
2004 00000B1C B020                          MOV    AL,EOI       ; END OF INTERRUPT
2005                                        ;CLI                ; DISABLE INTERRUPTS TILL STACK CLEARED
2006 00000B1E E6A0                          OUT    INTB00,AL    ; FOR CONTROLLER #2
2007
2008                                        ; 23/05/2016
2009 00000B20 B020                          MOV    AL,EOI       ; GET END OF INTERRUPT MASK
2010 00000B22 FA                            CLI                 ; DISABLE INTERRUPTS TILL STACK CLEARED
2011 00000B23 E620                          OUT    INTA00,AL    ; END OF INTERRUPT TO 8259 - 1
2012                                        ;
2013                                        ; 23/05/2016
2014 00000B25 20E4                          and    ah, ah
2015 00000B27 0F84B7FEFFFF                    jz      rtc_int_2
2016
2017                                        ; ah = 1 (half second)
2018 00000B2D 58                            pop    eax ; *
2019 00000B2E 1F                            pop    ds  ; **
2020 00000B2F CF                            iretd
2021
2022                                ; ////////////////
2023
2024                                        ; 28/08/2014
2025                                irq0:
2026 00000B30 6A00                            push       dword 0
2027 00000B32 EB48                          jmp    short which_irq
2028                                irq1:
2029 00000B34 6A01                            push       dword 1
2030 00000B36 EB44                          jmp    short which_irq
2031                                irq2:
2032 00000B38 6A02                            push       dword 2
2033 00000B3A EB40                          jmp    short which_irq
2034                                irq3:
2035                                        ; 20/11/2015
2036                                        ; 24/10/2015
2037 00000B3C 2EFF15[3A270100]              call   dword [cs:com2_irq3]
2038 00000B43 6A03                          push   dword 3
2039 00000B45 EB35                          jmp    short which_irq
2040                                irq4:
2041                                        ; 20/11/2015
2042                                        ; 24/10/2015
2043 00000B47 2EFF15[36270100]              call   dword [cs:com1_irq4]
2044 00000B4E 6A04                            push       dword 4
2045 00000B50 EB2A                          jmp    short which_irq
2046                                irq5:
2047 00000B52 6A05                            push       dword 5
2048 00000B54 EB26                          jmp    short which_irq
2049                                irq6:
2050 00000B56 6A06                            push       dword 6
2051 00000B58 EB22                          jmp    short which_irq
2052                                irq7:
2053 00000B5A 6A07                            push       dword 7
2054 00000B5C EB1E                          jmp    short which_irq
2055                                irq8:
2056 00000B5E 6A08                            push       dword 8
2057 00000B60 EB1A                          jmp    short which_irq
2058                                irq9:
2059 00000B62 6A09                            push       dword 9
2060 00000B64 EB16                          jmp    short which_irq
2061                                irq10:
2062 00000B66 6A0A                            push       dword 10
2063 00000B68 EB12                          jmp    short which_irq
2064                                irq11:
2065 00000B6A 6A0B                            push       dword 11
2066 00000B6C EB0E                          jmp    short which_irq
2067                                irq12:
2068 00000B6E 6A0C                            push       dword 12
2069 00000B70 EB0A                          jmp    short which_irq
2070                                irq13:
2071 00000B72 6A0D                            push       dword 13
2072 00000B74 EB06                          jmp    short which_irq
```

```
                                irq14:
00000B76 6A0E                           push       dword 14
00000B78 EB02                         jmp    short which_irq
                                irq15:
00000B7A 6A0F                           push       dword 15
                                      ;jmp   short which_irq

                                      ; 22/01/2017
                                      ; 19/10/2015
                                      ; 29/08/2014
                                      ; 21/08/2014
                                which_irq:
00000B7C 870424                       xchg   eax, [esp]  ; 28/08/2014
00000B7F 53                           push   ebx
00000B80 56                           push   esi
00000B81 57                           push   edi
00000B82 1E                           push   ds
00000B83 06                           push   es
                                      ;
00000B84 88C3                         mov    bl, al
                                      ;
00000B86 B810000000                   mov    eax, KDATA
00000B8B 8ED8                         mov    ds, ax
00000B8D 8EC0                         mov    es, ax
                                      ; 19/10/2015
00000B8F FC                           cld
                                        ; 27/08/2014
00000B90 8105[98410100]A000-            add     dword [scr_row], 0A0h
00000B98 0000
                                      ;
00000B9A B417                         mov    ah, 17h      ; blue (1) background,
                                                ; light gray (7) forecolor
00000B9C 8B3D[98410100]                 mov     edi, [scr_row]
00000BA2 B049                         mov    al, 'I'
00000BA4 66AB                         stosw
00000BA6 B052                         mov    al, 'R'
00000BA8 66AB                         stosw
00000BAA B051                         mov    al, 'Q'
00000BAC 66AB                         stosw
00000BAE B020                         mov    al, ' '
00000BB0 66AB                         stosw
00000BB2 88D8                         mov    al, bl
00000BB4 3C0A                         cmp    al, 10
00000BB6 7208                         jb     short ii1
00000BB8 B031                         mov    al, '1'
00000BBA 66AB                         stosw
00000BBC 88D8                         mov    al, bl
00000BBE 2C0A                         sub    al, 10
                                ii1:
00000BC0 0430                         add    al, '0'
00000BC2 66AB                         stosw
00000BC4 B020                         mov    al, ' '
00000BC6 66AB                         stosw
00000BC8 B021                         mov    al, '!'
00000BCA 66AB                         stosw
00000BCC B020                         mov    al, ' '
00000BCE 66AB                         stosw
                                      ; 23/02/2015
00000BD0 80FB07                       cmp    bl, 7 ; check for IRQ 8 to IRQ 15
00000BD3 7604                         jna    ii2
                                      ; 22/01/2017
00000BD5 B020                         mov    al, 20h  ; END OF INTERRUPT COMMAND TO
00000BD7 E6A0                         out    0A0h, al ; the 2nd 8259
                                ii2:
00000BD9 B020                         mov    al, 20h  ; END OF INTERRUPT COMMAND TO
00000BDB E620                         out    20h, al ; the 2nd 8259
00000BDD E9CD010000                   jmp     iiret
                                      ;
                                      ; 22/08/2014
                                      ;mov   al, 20h ; END OF INTERRUPT COMMAND TO 8259
                                      ;out   20h, al      ; 8259 PORT
                                      ;
                                      ;pop   es
                                      ;pop   ds
                                      ;pop   edi
                                      ;pop   esi
                                      ;pop   ebx
                                      ;pop   eax
                                      ;iret

                                      ; 02/04/2015
                                      ; 25/08/2014
                              exc0:
00000BE2 6A00                           push       dword 0
00000BE4 E990000000                     jmp     cpu_except
                              exc1:
00000BE9 6A01                           push       dword 1
00000BEB E989000000                     jmp     cpu_except
                              exc2:
00000BF0 6A02                           push       dword 2
00000BF2 E982000000                     jmp     cpu_except
                              exc3:
00000BF7 6A03                           push       dword 3
00000BF9 EB7E                           jmp     cpu_except
                              exc4:
00000BFB 6A04                           push       dword 4
00000BFD EB7A                           jmp     cpu_except
                              exc5:
00000BFF 6A05                           push       dword 5
00000C01 EB76                           jmp     cpu_except
                              exc6:
00000C03 6A06                           push       dword 6
00000C05 EB72                           jmp     cpu_except
                              exc7:
00000C07 6A07                           push       dword 7
```

```
2177 00000C09 EB6E                      jmp     cpu_except
2178                                exc8:
2179                                    ; [esp] = Error code
2180 00000C0B 6A08                      push    dword 8
2181 00000C0D EB5C                      jmp     cpu_except_en
2182                                exc9:
2183 00000C0F 6A09                      push    dword 9
2184 00000C11 EB66                      jmp     cpu_except
2185                                exc10:
2186                                    ; [esp] = Error code
2187 00000C13 6A0A                      push    dword 10
2188 00000C15 EB54                      jmp     cpu_except_en
2189                                exc11:
2190                                    ; [esp] = Error code
2191 00000C17 6A0B                      push    dword 11
2192 00000C19 EB50                      jmp     cpu_except_en
2193                                exc12:
2194                                    ; [esp] = Error code
2195 00000C1B 6A0C                      push    dword 12
2196 00000C1D EB4C                      jmp     cpu_except_en
2197                                exc13:
2198                                    ; [esp] = Error code
2199 00000C1F 6A0D                      push    dword 13
2200 00000C21 EB48                      jmp     cpu_except_en
2201                                exc14:
2202                                    ; [esp] = Error code
2203 00000C23 6A0E                      push    dword 14
2204 00000C25 EB44                    jmp   short cpu_except_en
2205                                exc15:
2206 00000C27 6A0F                      push    dword 15
2207 00000C29 EB4E                      jmp     cpu_except
2208                                exc16:
2209 00000C2B 6A10                      push    dword 16
2210 00000C2D EB4A                      jmp     cpu_except
2211                                exc17:
2212                                    ; [esp] = Error code
2213 00000C2F 6A11                      push    dword 17
2214 00000C31 EB38                    jmp   short cpu_except_en
2215                                exc18:
2216 00000C33 6A12                      push    dword 18
2217 00000C35 EB42                    jmp   short cpu_except
2218                                exc19:
2219 00000C37 6A13                      push    dword 19
2220 00000C39 EB3E                    jmp   short cpu_except
2221                                exc20:
2222 00000C3B 6A14                      push    dword 20
2223 00000C3D EB3A                    jmp   short cpu_except
2224                                exc21:
2225 00000C3F 6A15                      push    dword 21
2226 00000C41 EB36                    jmp   short cpu_except
2227                                exc22:
2228 00000C43 6A16                      push    dword 22
2229 00000C45 EB32                    jmp   short cpu_except
2230                                exc23:
2231 00000C47 6A17                      push    dword 23
2232 00000C49 EB2E                    jmp   short cpu_except
2233                                exc24:
2234 00000C4B 6A18                      push    dword 24
2235 00000C4D EB2A                    jmp   short cpu_except
2236                                exc25:
2237 00000C4F 6A19                      push    dword 25
2238 00000C51 EB26                    jmp   short cpu_except
2239                                exc26:
2240 00000C53 6A1A                      push    dword 26
2241 00000C55 EB22                    jmp   short cpu_except
2242                                exc27:
2243 00000C57 6A1B                      push    dword 27
2244 00000C59 EB1E                    jmp   short cpu_except
2245                                exc28:
2246 00000C5B 6A1C                      push    dword 28
2247 00000C5D EB1A                    jmp   short cpu_except
2248                                exc29:
2249 00000C5F 6A1D                      push    dword 29
2250 00000C61 EB16                    jmp   short cpu_except
2251                                exc30:
2252 00000C63 6A1E                      push    dword 30
2253 00000C65 EB04                    jmp   short cpu_except_en
2254                                exc31:
2255 00000C67 6A1F                      push    dword 31
2256 00000C69 EB0E                      jmp   short cpu_except
2257
2258                                    ; 19/10/2015
2259                                    ; 19/09/2015
2260                                    ; 01/09/2015
2261                                    ; 28/08/2015
2262                                    ; 28/08/2014
2263                                cpu_except_en:
2264 00000C6B 87442404                  xchg  eax, [esp+4] ; Error code
2265 00000C6F 36A3[78050300]            mov   [ss:error_code], eax
2266 00000C75 58                        pop   eax  ; Exception number
2267 00000C76 870424                    xchg  eax, [esp]
2268                                            ; eax = eax before exception
2269                                            ; [esp] -> exception number
2270                                            ; [esp+4] -> EIP to return
2271                                    ; 22/01/2017
2272                                    ; 19/10/2015
2273                                    ; 19/09/2015
2274                                    ; 01/09/2015
2275                                    ; 28/08/2015
2276                                    ; 29/08/2014
2277                                    ; 28/08/2014
2278                                    ; 25/08/2014
2279                                    ; 21/08/2014
2280                                cpu_except: ; CPU Exceptions
2281 00000C79 FC                        cld
```

```
2282 00000C7A 870424                    xchg   eax, [esp]
2283                                    ; eax = Exception number
2284                                    ; [esp] = eax (before exception)
2285 00000C7D 53                        push   ebx
2286 00000C7E 56                        push   esi
2287 00000C7F 57                        push   edi
2288 00000C80 1E                        push   ds
2289 00000C81 06                        push   es
2290                                    ; 28/08/2015
2291 00000C82 66BB1000                  mov    bx, KDATA
2292 00000C86 8EDB                      mov    ds, bx
2293 00000C88 8EC3                      mov    es, bx
2294 00000C8A 0F20DB                    mov    ebx, cr3
2295 00000C8D 53                        push   ebx ; (*) page directory
2296                                    ; 19/10/2015
2297 00000C8E FC                        cld
2298                                    ; 25/03/2015
2299 00000C8F 8B1D[90810100]            mov    ebx, [k_page_dir]
2300 00000C95 0F22DB                    mov    cr3, ebx
2301                                    ; 28/08/2015
2302 00000C98 83F80E                    cmp    eax, 0Eh ; 14, PAGE FAULT
2303 00000C9B 750F                      jne    short cpu_except_nfp
2304 00000C9D E8FF510000                call   page_fault_handler
2305 00000CA2 21C0                      and    eax, eax
2306 00000CA4 0F8401010000                jz   iiretp ; 01/09/2015
2307 00000CAA B00E                      mov    al, 0Eh ; 14
2308                                cpu_except_nfp:
2309                                    ; 23/08/2016
2310 00000CAC 803D[BA6F0000]03          cmp    byte [CRT_MODE], 3
2311 00000CB3 7409                      je     short cpu_except_mode_3
2312 00000CB5 50                        push   eax
2313 00000CB6 B003                      mov    al, 3
2314 00000CB8 E8AB0E0000                call    set_mode
2315 00000CBD 58                        pop    eax
2316                                cpu_except_mode_3:
2317                                    ; 02/04/2015
2318 00000CBE BB[0B090000]              mov    ebx, hang
2319 00000CC3 875C241C                  xchg   ebx, [esp+28]
2320                                    ; EIP (points to instruction which faults)
2321                                    ; New EIP (hang)
2322 00000CC7 891D[7C050300]            mov    [FaultOffset], ebx
2323 00000CCD C744242008000000          mov    dword [esp+32], KCODE ; kernel's code segment
2324 00000CD5 814C242400020000          or     dword [esp+36], 200h ; enable interrupts (set IF)
2325                                    ;
2326 00000CDD 88C4                      mov    ah, al
2327 00000CDF 240F                      and    al, 0Fh
2328 00000CE1 3C09                      cmp    al, 9
2329 00000CE3 7602                      jna    short h1ok
2330 00000CE5 0407                      add    al, 'A'-':'
2331                                h1ok:
2332 00000CE7 C0EC04                    shr    ah, 4
2333 00000CEA 80FC09                    cmp    ah, 9
2334 00000CED 7603                      jna    short h2ok
2335 00000CEF 80C407                    add    ah, 'A'-':'
2336                                h2ok:
2337 00000CF2 86E0                      xchg   ah, al
2338 00000CF4 66053030                  add    ax, '00'
2339 00000CF8 66A3[F0430100]            mov    [excnstr], ax
2340                                    ;
2341                                    ; 29/08/2014
2342 00000CFE A1[7C050300]              mov    eax, [FaultOffset]
2343 00000D03 51                        push   ecx
2344 00000D04 52                        push   edx
2345 00000D05 89E3                      mov    ebx, esp
2346                                    ; 28/08/2015
2347 00000D07 B910000000                mov    ecx, 16        ; divisor value to convert binary number
2348                                                       ; to hexadecimal string
2349                                    ;mov   ecx, 10          ; divisor to convert
2350                                                         ; binary number to decimal string
2351                                b2d1:
2352 00000D0C 31D2                      xor    edx, edx
2353 00000D0E F7F1                      div    ecx
2354 00000D10 6652                      push   dx
2355 00000D12 39C8                      cmp    eax, ecx
2356 00000D14 73F6                      jnb    short b2d1
2357 00000D16 BF[FB430100]              mov    edi, EIPstr ; EIP value
2358                                                       ; points to instruction which faults
2359                                    ; 28/08/2015
2360 00000D1B 89C2                      mov    edx, eax
2361                                b2d2:
2362                                    ;add   al, '0'
2363 00000D1D 8A82[15430000]            mov    al, [edx+hexchrs]
2364 00000D23 AA                        stosb             ; write hexadecimal digit to its place
2365 00000D24 39E3                      cmp    ebx, esp
2366 00000D26 7606                      jna    short b2d3
2367 00000D28 6658                      pop    ax
2368 00000D2A 88C2                      mov    dl, al
2369 00000D2C EBEF                      jmp    short b2d2
2370                                b2d3:
2371 00000D2E B068                      mov    al, 'h' ; 28/08/2015
2372 00000D30 AA                        stosb
2373 00000D31 B020                      mov    al, 20h          ; space
2374 00000D33 AA                        stosb
2375 00000D34 30C0                      xor    al, al     ; to do it an ASCIIZ string
2376 00000D36 AA                        stosb
2377                                    ;
2378 00000D37 5A                        pop    edx
2379 00000D38 59                        pop    ecx
2380                                    ;
2381 00000D39 B44F                      mov    ah, 4Fh      ; red (4) background,
2382                                                  ; white (F) forecolor
2383 00000D3B BE[E0430100]              mov    esi, exc_msg ; message offset
2384                                    ;
2385                                    ; 20/01/2017 (!cpu exception!)
2386                                    ;
```

```
2387 00000D40 8105[98410100]A000-           add    dword [scr_row], 0A0h
2387 00000D48 0000
2388 00000D4A 8B3D[98410100]                mov    edi, [scr_row]
2389                                        ;
2390 00000D50 C605[5B030300]00              mov    byte [sysflg], 0  ; system mode
2391 00000D57 FB                              sti
2392                                        ;
2393 00000D58 E8EDFBFFFF                    call   printk
2394                                        ;
2395 00000D5D B410                          mov    ah, 10h
2396 00000D5F E881010000                    call   int16h ; getc
2397                                        ;
2398 00000D64 B003                          mov    al, 3
2399 00000D66 E8FD0D0000                    call   _set_mode
2400                                        ;
2401 00000D6B B801000000                    mov    eax, 1
2402 00000D70 E9FECC0000                    jmp    sysexit ; terminate process !!!
2403
2404                                        ; 22/01/2017
2405                                        ; 18/04/2016
2406                                        ; 28/08/2015
2407                                        ; 23/02/2015
2408                                        ; 20/08/2014
2409                                  ignore_int:
2410 00000D75 50                            push   eax
2411 00000D76 53                            push   ebx ; 23/02/2015
2412 00000D77 56                            push   esi
2413 00000D78 57                            push   edi
2414 00000D79 1E                            push   ds
2415 00000D7A 06                            push   es
2416                                        ; 18/04/2016
2417 00000D7B 66B81000                      mov    ax, KDATA
2418 00000D7F 8ED8                          mov    ds, ax
2419 00000D81 8EC0                          mov    es, ax
2420                                        ; 28/08/2015
2421 00000D83 0F20D8                        mov    eax, cr3
2422 00000D86 50                            push   eax ; (*) page directory
2423                                        ;
2424 00000D87 B467                          mov    ah, 67h      ; brown (6) background,
2425                                                 ; light gray (7) forecolor
2426 00000D89 BE[A8420100]                  mov    esi, int_msg ; message offset
2427                                  piemsg:
2428                                          ; 27/08/2014
2429 00000D8E 8105[98410100]A000-             add    dword [scr_row], 0A0h
2429 00000D96 0000
2430 00000D98 8B3D[98410100]                  mov    edi, [scr_row]
2431                                          ;
2432 00000D9E E8A7FBFFFF                    call   printk
2433                                        ;
2434                                        ; 23/02/2015
2435 00000DA3 B020                          mov    al, 20h  ; END OF INTERRUPT COMMAND TO
2436 00000DA5 E6A0                          out    0A0h, al ; the 2nd 8259
2437                                        ; 22/08/2014
2438 00000DA7 B020                          mov    al, 20h ; END OF INTERRUPT COMMAND TO 8259
2439 00000DA9 E620                          out    20h, al      ; 8259 PORT
2440                                  iiretp:
2441                                        ; 22/01/2017
2442                                        ; 01/09/2015
2443                                        ; 28/08/2015
2444 00000DAB 58                            pop    eax ; (*) page directory
2445 00000DAC 0F22D8                        mov    cr3, eax
2446                                  iiret:
2447 00000DAF 07                            pop    es
2448 00000DB0 1F                            pop    ds
2449 00000DB1 5F                            pop    edi
2450 00000DB2 5E                            pop    esi
2451 00000DB3 5B                            pop    ebx ; 29/08/2014
2452 00000DB4 58                            pop    eax
2453 00000DB5 CF                            iretd
2454
2455                                        ; 23/05/2016
2456                                        ; 22/08/2014
2457                                        ; IBM PC/AT BIOS source code ----- 10/06/85 (bios.asm)
2458                                        ; (INT 1Ah)
2459                                        ;; Linux (v0.12) source code (main.c) by Linus Torvalds (1991)
2460                                  time_of_day:
2461 00000DB6 E8D55E0000                    call   UPD_IPR                    ; WAIT TILL UPDATE NOT IN PROGRESS
2462 00000DBB 726F                            jc     short time_of_day_retn ; 23/05/2016
2463 00000DBD B000                          mov    al, CMOS_SECONDS
2464 00000DBF E8E75E0000                    call   CMOS_READ
2465 00000DC4 A2[00820100]                  mov    [time_seconds], al
2466 00000DC9 B002                          mov    al, CMOS_MINUTES
2467 00000DCB E8DB5E0000                    call   CMOS_READ
2468 00000DD0 A2[01820100]                  mov    [time_minutes], al
2469 00000DD5 B004                          mov    al, CMOS_HOURS
2470 00000DD7 E8CF5E0000                    call   CMOS_READ
2471 00000DDC A2[02820100]                    mov    [time_hours], al
2472 00000DE1 B006                          mov    al, CMOS_DAY_WEEK
2473 00000DE3 E8C35E0000                    call   CMOS_READ
2474 00000DE8 A2[03820100]                  mov    [date_wday], al
2475 00000DED B007                          mov    al, CMOS_DAY_MONTH
2476 00000DEF E8B75E0000                    call   CMOS_READ
2477 00000DF4 A2[04820100]                  mov    [date_day], al
2478 00000DF9 B008                          mov    al, CMOS_MONTH
2479 00000DFB E8AB5E0000                    call   CMOS_READ
2480 00000E00 A2[05820100]                  mov    [date_month], al
2481 00000E05 B009                          mov    al, CMOS_YEAR
2482 00000E07 E89F5E0000                    call   CMOS_READ
2483 00000E0C A2[06820100]                  mov    [date_year], al
2484 00000E11 B032                          mov    al, CMOS_CENTURY
2485 00000E13 E8935E0000                    call   CMOS_READ
2486 00000E18 A2[07820100]                  mov    [date_century], al
2487                                        ;
2488 00000E1D B000                          mov    al, CMOS_SECONDS
2489 00000E1F E8875E0000                    call   CMOS_READ
```

```
2490 00000E24 3A05[00820100]              cmp   al, [time_seconds]
2491 00000E2A 758A                        jne   short time_of_day
2492
2493                                  time_of_day_retn:
2494 00000E2C C3                          retn
2495
2496                                      ; 15/01/2017
2497                                      ; 10/06/2016
2498                                      ; 07/06/2016
2499                                      ; 06/06/2016
2500                                      ; 23/05/2016
2501                                  rtc_p:
2502 00000E2D B101                        mov   cl, 1 ; 15/01/2017
2503 00000E2F EB02                        jmp   short rtc_p0
2504                                  u_timer:
2505                                      ; Timer Events with 18.2 Hz Timer Ticks
2506                                      ; (and also timer events with RTC seconds)
2507 00000E31 28C9                        sub   cl, cl ; mov cl, 0 ; 15/01/2017
2508                                  rtc_p0:
2509                                      ; 19/05/2016 - TRDOS 386 (TRDOS v2.0)
2510                                      ; Major Modification:
2511                                      ; Check and Perform Timer Events (for RTC)
2512                                      ; 25/08/2014 - 07/09/2014
2513                                      ; Retro UNIX 386 v1:
2514                                      ; Print Real Time Clock content
2515
2516                                      ; 15/01/2017
2517 00000E33 880D[208E0100]              mov   byte [priority], cl ; 0 or 1 (not 2)
2518 00000E39 8A2D[238E0100]              mov   ch, [timer_events]
2519 00000E3F 20ED                        and   ch, ch
2520 00000E41 7420                        jz    short rtc_p3
2521
2522 00000E43 BE[60040300]                mov   esi, timer_set  ; beginning address of
2523                                                             ; timer events space
2524                                  rtc_p1:
2525 00000E48 8B06                        mov   eax, [esi]
2526 00000E4A 20C0                        and   al, al ; 0 = free, >0 = process no.
2527 00000E4C 7416                        jz    short rtc_p4
2528                                      ;
2529 00000E4E C1C810                      ror   eax, 16
2530                                      ; ah = response value, al = interrupt type
2531                                      ; 15/01/2017
2532                                      ; cl = interrupt source
2533                                      ;       1 = RTC, 0 = PIT
2534 00000E51 38C8                        cmp   al, cl
2535 00000E53 750A                        jne   short rtc_p2 ; not as requested or undefined !
2536 00000E55 3C01                        cmp   al, 1 ; 1 ; RTC interrupt ?
2537 00000E57 7410                        je    short rtc_p5 ; yes, check for response
2538                                      ; 06/06/2016 - 18.2 Hz Timer Ticks
2539 00000E59 836E080A                    sub   dword [esi+8], 10 ; 1 tick = 10
2540 00000E5D 7613                        jna   short rtc_p6  ; continue for responding
2541                                  rtc_p2:
2542                                      ; 15/01/2017 (cl -> ch)
2543                                      ; 07/06/2016
2544 00000E5F FECD                        dec   ch    ; remain count of timer events
2545 00000E61 7501                        jnz   short rtc_p4
2546                                  rtc_p3:
2547 00000E63 C3                          retn
2548                                  rtc_p4:
2549                                      ;cmp   esi, timer_set + 240 ; 15*16 (last event)
2550                                      ;jnb   short rtc_p3 ; end of timer event space
2551 00000E64 83C610                      add   esi, 16 ; next timer event
2552 00000E67 EBDF                        jmp   short rtc_p1
2553                                  rtc_p5:
2554                                      ; current timer count ; 06/06/2016 (182)
2555 00000E69 816E08B6000000              sub   dword [esi+8], 182 ; 1 second (10*18.2)
2556 00000E70 77ED                        ja    short rtc_p2  ; check for the next
2557                                  rtc_p6:
2558                                      ; it is the time of response!
2559 00000E72 8B5E04                      mov   ebx, [esi+4] ; set (count limit) value
2560 00000E75 895E08                      mov   [esi+8], ebx ; reset count down value
2561                                                         ; to count limit
2562                                      ; 19/12/2016
2563                                      ; 10/12/2016 - timer callback modification
2564 00000E78 8B7E0C                      mov   edi, [esi+12] ; response (or callback) address
2565 00000E7B 807E0100                    cmp   byte [esi+1], 0 ; >0 = callback
2566 00000E7F 762A                        jna   short rtc_p8
2567
2568                                      ; timer callback !
2569 00000E81 0FB61E                      movzx  ebx, byte [esi] ; process number (>0)
2570 00000E84 89D8                        mov   eax, ebx
2571 00000E86 C0E302                      shl   bl, 2 ; *4
2572 00000E89 89BB[0C010300]              mov   [ebx+p.tcb-4], edi ; user's callback service addr
2573 00000E8F 3A05[B3030300]              cmp   al, [u.uno]
2574 00000E95 7521                        jne   short rtc_p9
2575 00000E97 893D[D0030300]              mov   [u.tcb], edi
2576                                  rtc_p7:
2577                                      ; 15/01/2017
2578 00000E9D B002                        mov   al, 2
2579 00000E9F A2[208E0100]                mov   [priority], al ; 2
2580                                      ; 10/01/2017
2581                                      ;mov  byte [u.pri], 2
2582 00000EA4 A2[A9030300]                mov   [u.pri], al ; 2
2583 00000EA9 EBB4                        jmp   short rtc_p2
2584                                  rtc_p8:
2585                                      ; response address is physical address of
2586                                      ; the program's response (signal return) byte
2587                                      ; 06/06/2016
2588                                      ;mov  edi, [esi+12] ; response address
2589 00000EAB 8827                        mov   [edi], ah     ; response value
2590                                      ;
2591 00000EAD C1C010                      rol   eax, 16
2592                                      ; 15/01/2017
2593 00000EB0 3A05[B3030300]              cmp   al, [u.uno] ; running process ?
2594 00000EB6 74E5                        je    short rtc_p7
```

```
                                rtc_p9:
                                      ; al = process number  ; 10/06/2016
00000EB8 B202                         mov   dl, 2 ; priority, 2 = event (high)
00000EBA E871150100                   call  set_run_sequence ; 19/05/2016
00000EBF 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:
00000EC1 6650                         push  ax
00000EC3 B00B                         mov   al, 0Bh  ; In-Service register
00000EC5 E620                         out   20h, al
00000EC7 EB00                           jmp short $+2
00000EC9 EB00                         jmp short $+2
00000ECB E420                         in    al, 20h
00000ECD 2480                         and   al, 80h ; bit 7 (is it real IRQ 7 or fake?)
00000ECF 7404                           jz      short irq7_iret ; Fake (spurious) IRQ, do not send EOI
00000ED1 B020                           mov     al, 20h ; EOI
00000ED3 E620                         out   20h, al
                                irq7_iret:
00000ED5 6658                         pop   ax
00000ED7 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)
                                      ;
00000ED8 D410                         db 0D4h,10h                   ; Undocumented inst. AAM
                                                                    ; AH = AL / 10h
                                                                    ; AL = AL MOD 10h
00000EDA 660D3030                   or ax,'00'                      ; Make it ASCII based

00000EDE 86E0                         xchg ah, al

00000EE0 C3                           retn

                                ; 15/12/2020
                                real_mem_16m_64k:
00000EE1 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:
00000EE3 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:
00000EE5 9C                     <1>       pushfd ; 28/08/2014
00000EE6 0E                     <1>       push   cs
00000EE7 E801000000             <1>       call   KEYBOARD_IO_1 ; getc_int
00000EEC 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
00000EED 80642408BE     <1>         and     byte [esp+8], 10111110b ; clear zero flag and cary flag
                        <1>       ;
00000EF2 1E             <1>       push  ds                ; SAVE CURRENT DS
00000EF3 53             <1>       push  ebx               ; SAVE BX TEMPORARILY
                        <1>       ;push ecx               ; SAVE CX TEMPORARILY
00000EF4 66BB1000       <1>         mov     bx, KDATA
00000EF8 8EDB           <1>       mov   ds, bx            ; PUT SEGMENT VALUE OF DATA AREA INTO DS
                        <1>
```

```
                        <1>       ; 14/01/2017
00000EFA 8B1C24         <1>       mov    ebx, [esp]
                        <1>       ;; 15/01/2017
                        <1>       ; 02/01/2017
                        <1>       ;;mov  byte [intflg], 32h  ; keyboard interrupt
00000EFD FB             <1>       sti
                        <1>       ;
                        <1>
00000EFE 08E4           <1>       or     ah, ah              ; CHECK FOR (AH)= 00H
00000F00 743A           <1>       jz     short _K1           ; ASCII_READ
00000F02 FECC           <1>       dec    ah                     ; CHECK FOR (AH)= 01H
00000F04 7453           <1>         jz       short _K2              ; ASCII_STATUS
00000F06 FECC           <1>       dec    ah                  ; CHECK FOR (AH)= 02H
00000F08 0F8494000000   <1>         jz       _K3                    ; SHIFT STATUS
00000F0E FECC           <1>       dec    ah                  ; CHECK FOR (AH)= 03H
00000F10 0F8493000000   <1>         jz       _K300                  ; SET TYPAMATIC RATE/DELAY
00000F16 80EC02         <1>       sub    ah, 2               ; CHECK FOR (AH)= 05H
00000F19 0F84BC000000   <1>         jz       _K500                  ; KEYBOARD WRITE
                        <1> _KIO1:
00000F1F 80EC0B         <1>       sub    ah, 11              ; AH =  10H
00000F22 740C           <1>       jz     short _K1E          ; EXTENDED ASCII READ
00000F24 FECC           <1>       dec    ah                  ; CHECK FOR (AH)= 11H
00000F26 7422           <1>       jz     short _K2E          ; EXTENDED_ASCII_STATUS
00000F28 FECC           <1>       dec    ah                  ; CHECK FOR (AH)= 12H
00000F2A 7458           <1>       jz     short _K3E          ; EXTENDED_SHIFT_STATUS
                        <1> _KIO_EXIT:
                        <1>       ; 02/01/2017
00000F2C FA             <1>       cli
                        <1>       ;;mov  byte [intflg], 0 ;; 15/01/2017
                        <1>       ;
                        <1>       ;pop   ecx                 ; RECOVER REGISTER
00000F2D 5B             <1>       pop    ebx                 ; RECOVER REGISTER
00000F2E 1F             <1>       pop    ds                  ; RECOVER SEGMENT
00000F2F CF             <1>       iretd                      ; INVALID COMMAND, EXIT
                        <1>
                        <1>       ;----- ASCII CHARACTER
                        <1> _K1E:
00000F30 E8D3000000     <1>       call   _K1S                ; GET A CHARACTER FROM THE BUFFER (EXTENDED)
00000F35 E848010000     <1>       call   _KIO_E_XLAT         ; ROUTINE TO XLATE FOR EXTENDED CALLS
00000F3A EBF0           <1>       jmp    short _KIO_EXIT         ; GIVE IT TO THE CALLER
                        <1> _K1:
00000F3C E8C7000000     <1>       call   _K1S                ; GET A CHARACTER FROM THE BUFFER
00000F41 E847010000     <1>       call   _KIO_S_XLAT         ; ROUTINE TO XLATE FOR STANDARD CALLS
00000F46 72F4           <1>       jc     short _K1           ; CARRY SET MEANS TROW CODE AWAY
                        <1> _K1A:
00000F48 EBE2           <1>       jmp    short _KIO_EXIT         ; RETURN TO CALLER
                        <1>
                        <1>       ;----- ASCII STATUS
                        <1> _K2E:
00000F4A E804010000     <1>       call   _K2S                ; TEST FOR CHARACTER IN BUFFER (EXTENDED)
00000F4F 7420           <1>       jz     short _K2B          ; RETURN IF BUFFER EMPTY
00000F51 9C             <1>       pushf                      ; SAVE ZF FROM TEST
00000F52 E82B010000     <1>       call   _KIO_E_XLAT         ; ROUTINE TO XLATE FOR EXTENDED CALLS
00000F57 EB17           <1>       jmp    short _K2A           ; GIVE IT TO THE CALLER
                        <1> _K2:
00000F59 E8F5000000     <1>       call   _K2S                ; TEST FOR CHARACTER IN BUFFER
00000F5E 7411           <1>       jz     short _K2B          ; RETURN IF BUFFER EMPTY
00000F60 9C             <1>       pushf                      ; SAVE ZF FROM TEST
00000F61 E827010000     <1>       call   _KIO_S_XLAT         ; ROUTINE TO XLATE FOR STANDARD CALLS
00000F66 7308           <1>       jnc    short _K2A           ; CARRY CLEAR MEANS PASS VALID CODE
00000F68 9D             <1>       popf                       ; INVALID CODE FOR THIS TYPE OF CALL
00000F69 E89A000000     <1>       call   _K1S                ; THROW THE CHARACTER AWAY
00000F6E EBE9           <1>       jmp    short _K2           ; GO LOOK FOR NEXT CHAR, IF ANY
                        <1> _K2A:
00000F70 9D             <1>       popf                       ; RESTORE ZF FROM TEST
                        <1> _K2B:
                        <1>       ; 02/01/2017
00000F71 FA             <1>       cli
                        <1>       ;; mov byte [intflg], 0 ;; 15/01/2017
                        <1>       ;
                        <1>       ;pop   ecx                 ; RECOVER REGISTER
00000F72 5B             <1>       pop    ebx                 ; RECOVER REGISTER
00000F73 1F             <1>       pop    ds                  ; RECOVER SEGMENT
                        <1>       ; (*) 29/05/2016
                        <1>       ; (*) retf 4               ; THROW AWAY (e)FLAGS
00000F74 7208           <1>       jc     short _k2d
00000F76 7505           <1>       jnz    short _k2c
00000F78 804C240840     <1>       or     byte [esp+8], 01000000b   ; set zero flag bit of eflags register
                        <1> _k2c:
00000F7D 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
00000F7E 804C240801     <1>       or     byte [esp+8], 1  ; set carry bit of eflags register
                        <1>       ; [esp+12] = ESP (user)
                        <1>       ; [esp+16] = SS (User)
00000F83 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
00000F84 8A25[866F0000] <1>       mov   ah, [KB_FLAG_1]           ; GET SYSTEM SHIFT KEY STATUS
00000F8A 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
00000F8D C0E405         <1>         shl ah, 5
00000F90 A0[866F0000]   <1>       mov   al, [KB_FLAG_1]           ; GET SYSTEM SHIFT STATES BACK
00000F95 2473           <1>       and   al, 01110011b       ; ELIMINATE SYS SHIFT, HOLD_STATE AND INS_SHIFT
00000F97 08C4           <1>       or    ah, al                 ; MERGE REMAINING BITS INTO AH
00000F99 A0[886F0000]   <1>       mov   al, [KB_FLAG_3]           ; GET RIGHT CTL AND ALT
00000F9E 240C           <1>       and   al, 00001100b       ; ELIMINATE LC_E0 AND LC_E1
00000FA0 08C4           <1>       or    ah, al              ; OR THE SHIFT FLAGS TOGETHER
                        <1> _K3:
00000FA2 A0[856F0000]   <1>       mov   al, [KB_FLAG]       ; GET THE SHIFT STATUS FLAGS
                        <1>       ;jmp  short _KIO_EXIT           ; RETURN TO CALLER
00000FA7 EB83           <1>       jmp   _KIO_EXIT
                        <1>
                        <1>       ;----- SET TYPAMATIC RATE AND DELAY
                        <1> _K300:
00000FA9 3C05           <1>       cmp   al, 5               ; CORRECT FUNCTION CALL?
                        <1>       ;jne  short _KIO_EXIT           ; NO, RETURN
00000FAB 0F857BFFFFFF   <1>       jne   _KIO_EXIT
00000FB1 F6C3E0         <1>       test  bl, 0E0h            ; TEST FOR OUT-OF-RANGE RATE
00000FB4 0F8572FFFFFF   <1>         jnz   _KIO_EXIT                 ; RETURN IF SO
00000FBA F6C7FC         <1>       test  BH, 0FCh            ; TEST FOR OUT-OF-RANGE DELAY
00000FBD 0F8569FFFFFF   <1>         jnz   _KIO_EXIT                 ; RETURN IF SO
00000FC3 B0F3           <1>       mov   al, KB_TYPA_RD            ; COMMAND FOR TYPAMATIC RATE/DELAY
00000FC5 E8DA060000     <1>       call  SND_DATA            ; SEND TO KEYBOARD
                        <1>       ;mov  cx, 5               ; SHIFT COUNT
                        <1>       ;shl  bh, cl              ; SHIFT DELAY OVER
00000FCA C0E705         <1>       shl   bh, 5
00000FCD 88D8           <1>       mov   al, bl              ; PUT IN RATE
00000FCF 08F8           <1>       or    al, bh              ; AND DELAY
00000FD1 E8CE060000     <1>       call  SND_DATA            ; SEND TO KEYBOARD
00000FD6 E951FFFFFF     <1>         jmp   _KIO_EXIT                 ; RETURN TO CALLER
                        <1>
                        <1>       ;----- WRITE TO KEYBOARD BUFFER
                        <1> _K500:
00000FDB 56             <1>       push  esi                 ; SAVE SI (esi)
00000FDC FA             <1>       cli                       ;
00000FDD 8B1D[966F0000] <1>       mov   ebx, [BUFFER_TAIL]  ; GET THE 'IN TO' POINTER TO THE BUFFER
00000FE3 89DE           <1>       mov   esi, ebx            ; SAVE A COPY IN CASE BUFFER NOT FULL
00000FE5 E8D3000000     <1>       call  _K4                 ; BUMP THE POINTER TO SEE IF BUFFER IS FULL
00000FEA 3B1D[926F0000] <1>       cmp   ebx, [BUFFER_HEAD]  ; WILL THE BUFFER OVERRUN IF WE STORE THIS?
00000FF0 740D           <1>       je    short _K502         ; YES - INFORM CALLER OF ERROR
00000FF2 66890E         <1>       mov   [esi], cx           ; NO - PUT ASCII/SCAN CODE INTO BUFFER
00000FF5 891D[966F0000] <1>       mov   [BUFFER_TAIL], ebx  ; ADJUST 'IN TO' POINTER TO REFLECT CHANGE
00000FFB 28C0           <1>       sub   al, al              ; TELL CALLER THAT OPERATION WAS SUCCESSFUL
00000FFD EB02           <1>       jmp   short _K504         ; SUB INSTRUCTION ALSO RESETS CARRY FLAG
                        <1> _K502:
00000FFF B001           <1>       mov   al, 01h                   ; BUFFER FULL INDICATION
                        <1> _K504:
00001001 FB             <1>       sti
00001002 5E             <1>       pop   esi                 ; RECOVER SI (esi)
00001003 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:
00001008 FA             <1>       cli   ; 03/12/2014
00001009 8B1D[926F0000] <1>         mov   ebx, [BUFFER_HEAD]    ; GET POINTER TO HEAD OF BUFFER
0000100F 3B1D[966F0000] <1>         cmp   ebx, [BUFFER_TAIL]    ; TEST END OF BUFFER
                        <1>       ;jne  short _K1U          ; IF ANYTHING IN BUFFER SKIP INTERRUPT
00001015 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
00001017 FB             <1>       sti                       ; INTERRUPTS BACK ON DURING LOOP
00001018 90             <1>       nop                       ; ALLOW AN INTERRUPT TO OCCUR
                        <1> _K1U:
00001019 FA             <1>       cli                       ; INTERRUPTS BACK OFF
0000101A 8B1D[926F0000] <1>         mov       ebx, [BUFFER_HEAD]  ; GET POINTER TO HEAD OF BUFFER
00001020 3B1D[966F0000] <1>         cmp     ebx, [BUFFER_TAIL]    ; TEST END OF BUFFER
                        <1> _k1x:
00001026 53             <1>       push  ebx                 ; SAVE ADDRESS
00001027 9C             <1>       pushf                     ; SAVE FLAGS
00001028 E82F070000     <1>       call  MAKE_LED            ; GO GET MODE INDICATOR DATA BYTE
0000102D 8A1D[876F0000] <1>       mov   bl, [KB_FLAG_2]     ; GET PREVIOUS BITS
00001033 30C3           <1>       xor   bl, al              ; SEE IF ANY DIFFERENT
00001035 80E307         <1>       and   bl, 07h      ; KB_LEDS    ; ISOLATE INDICATOR BITS
00001038 7406           <1>       jz    short _K1V          ; IF NO CHANGE BYPASS UPDATE
0000103A E8C9060000     <1>       call  SND_LED1
0000103F FA             <1>       cli                       ; DISABLE INTERRUPTS
                        <1> _K1V:
00001040 9D             <1>       popf                      ; RESTORE FLAGS
00001041 5B             <1>       pop   ebx                 ; RESTORE ADDRESS
00001042 74D3           <1>         je      short _K1T              ; LOOP UNTIL SOMETHING IN BUFFER
                        <1>       ;
00001044 668B03         <1>       mov   ax, [ebx]           ; GET SCAN CODE AND ASCII CODE
00001047 E871000000     <1>         call    _K4                     ; MOVE POINTER TO NEXT POSITION
0000104C 891D[926F0000] <1>         mov     [BUFFER_HEAD], ebx      ; STORE VALUE IN VARIABLE
00001052 C3             <1>       retn                      ; RETURN
                        <1>
                        <1>       ;----- READ THE KEY TO SEE IF ONE IS PRESENT -----
                        <1> _K2S:
00001053 FA             <1>       cli                       ; INTERRUPTS OFF
```

```
363 00001054 8B1D[926F0000]      <1>         mov     ebx, [BUFFER_HEAD]      ; GET HEAD POINTER
364 0000105A 3B1D[966F0000]      <1>         cmp     ebx, [BUFFER_TAIL]      ; IF EQUAL (Z=1) THEN NOTHING THERE
365 00001060 668B03              <1>       mov   ax, [ebx]
366 00001063 9C                  <1>       pushf                     ; SAVE FLAGS
367 00001064 6650                <1>       push  ax                  ; SAVE CODE
368 00001066 E8F1060000          <1>       call  MAKE_LED            ; GO GET MODE INDICATOR DATA BYTE
369 0000106B 8A1D[876F0000]      <1>       mov   bl, [KB_FLAG_2]     ; GET PREVIOUS BITS
370 00001071 30C3                <1>       xor   bl, al              ; SEE IF ANY DIFFERENT
371 00001073 80E307              <1>       and   bl, 07h ; KB_LEDS   ; ISOLATE INDICATOR BITS
372 00001076 7405                <1>       jz    short _K2T          ; IF NO CHANGE BYPASS UPDATE
373 00001078 E874060000          <1>       call  SND_LED                   ; GO TURN ON MODE INDICATORS
374                              <1> _K2T:
375 0000107D 6658                <1>       pop   ax                  ; RESTORE CODE
376 0000107F 9D                  <1>       popf                      ; RESTORE FLAGS
377 00001080 FB                  <1>       sti                       ; INTERRUPTS BACK ON
378 00001081 C3                  <1>       retn                      ; RETURN
379                              <1>
380                              <1>       ;----- ROUTINE TO TRANSLATE SCAN CODE PAIRS FOR EXTENDED CALLS -----
381                              <1> _KIO_E_XLAT:
382 00001082 3CF0                <1>       cmp   al, 0F0h            ; IS IT ONE OF THE FILL-INs?
383 00001084 7506                <1>       jne   short _KIO_E_RET    ; NO, PASS IT ON
384 00001086 08E4                <1>         or  ah, ah              ; AH = 0 IS SPECIAL CASE
385 00001088 7402                <1>         jz  short _KIO_E_RET        ; PASS THIS ON UNCHANGED
386 0000108A 30C0                <1>       xor   al, al              ; OTHERWISE SET AL = 0
387                              <1> _KIO_E_RET:
388 0000108C C3                  <1>       retn                      ; GO BACK
389                              <1>
390                              <1>       ;----- ROUTINE TO TRANSLATE SCAN CODE PAIRS FOR STANDARD CALLS -----
391                              <1> _KIO_S_XLAT:
392 0000108D 80FCE0              <1>       cmp   ah, 0E0h            ; IS IT KEYPAD ENTER OR / ?
393 00001090 750F                <1>       jne   short _KIO_S2       ; NO, CONTINUE
394 00001092 3C0D                <1>       cmp   al, 0Dh                   ; KEYPAD ENTER CODE?
395 00001094 7408                <1>         je  short _KIO_S1       ; YES, MASSAGE A BIT
396 00001096 3C0A                <1>       cmp   al, 0Ah                   ; CTRL KEYPAD ENTER CODE?
397 00001098 7404                <1>         je  short _KIO_S1       ; YES, MASSAGE THE SAME
398 0000109A B435                <1>       mov   ah, 35h                   ; NO, MUST BE KEYPAD /
399                              <1> _kio_ret: ; 03/12/2014
400 0000109C F8                  <1>       clc
401 0000109D C3                  <1>       retn
402                              <1>       ;jmp  short _KIO_USE            ; GIVE TO CALLER
403                              <1> _KIO_S1:
404 0000109E B41C                <1>       mov   ah, 1Ch                   ; CONVERT TO COMPATIBLE OUTPUT
405                              <1>       ;jmp  short _KIO_USE            ; GIVE TO CALLER
406 000010A0 C3                  <1>       retn
407                              <1> _KIO_S2:
408 000010A1 80FC84              <1>       cmp   ah, 84h                   ; IS IT ONE OF EXTENDED ONES?
409 000010A4 7715                <1>       ja    short _KIO_DIS            ; YES, THROW AWAY AND GET ANOTHER CHAR
410 000010A6 3CF0                <1>       cmp   al, 0F0h            ; IS IT ONE OF THE FILL-INs?
411 000010A8 7506                <1>         jne short _KIO_S3       ; NO, TRY LAST TEST
412 000010AA 08E4                <1>       or    ah, ah              ; AH = 0 IS SPECIAL CASE
413 000010AC 740C                <1>         jz  short _KIO_USE            ; PASS THIS ON UNCHANGED
414 000010AE EB0B                <1>       jmp   short _KIO_DIS            ; THROW AWAY THE REST
415                              <1> _KIO_S3:
416 000010B0 3CE0                <1>       cmp   al, 0E0h            ; IS IT AN EXTENSION OF A PREVIOUS ONE?
417                              <1>       ;jne  short _KIO_USE            ; NO, MUST BE A STANDARD CODE
418 000010B2 75E8                <1>       jne   short _kio_ret
419 000010B4 08E4                <1>       or    ah, ah              ; AH = 0 IS SPECIAL CASE
420 000010B6 7402                <1>         jz  short _KIO_USE            ; JUMP IF AH = 0
421 000010B8 30C0                <1>       xor   al, al              ; CONVERT TO COMPATIBLE OUTPUT
422                              <1>       ;jmp  short _KIO_USE            ; PASS IT ON TO CALLER
423                              <1> _KIO_USE:
424                              <1>       ;clc                      ; CLEAR CARRY TO INDICATE GOOD CODE
425 000010BA C3                  <1>       retn                      ; RETURN
426                              <1> _KIO_DIS:
427 000010BB F9                  <1>       stc                       ; SET CARRY TO INDICATE DISCARD CODE
428 000010BC C3                  <1>       retn                      ; RETURN
429                              <1>
430                              <1>       ;----- INCREMENT BUFFER POINTER ROUTINE -----
431                              <1> _K4:
432 000010BD 43                  <1>       inc     ebx
433 000010BE 43                  <1>       inc   ebx                 ; MOVE TO NEXT WORD IN LIST
434 000010BF 3B1D[8E6F0000]      <1>         cmp     ebx, [BUFFER_END]       ; AT END OF BUFFER?
435                              <1>         ;jne    short _K5               ; NO, CONTINUE
436 000010C5 7206                <1>       jb    short _K5
437 000010C7 8B1D[8A6F0000]      <1>         mov     ebx, [BUFFER_START]     ; YES, RESET TO BUFFER BEGINNING
438                              <1> _K5:
439 000010CD C3                  <1>       retn
440                              <1>
441                              <1> ; 20/02/2015
442                              <1> ; 05/12/2014
443                              <1> ; 26/08/2014
444                              <1> ; KEYBOARD (HARDWARE) INTERRUPT -  IRQ LEVEL 1
445                              <1> ; (INT_09h - Retro UNIX 8086 v1 - U9.ASM, 07/03/2014)
446                              <1> ;
447                              <1> ; Derived from "KB_INT_1" procedure of IBM "pc-at"
448                              <1> ; rombios source code (06/10/1985)
449                              <1> ; 'keybd.asm', HARDWARE INT 09h - (IRQ Level 1)
450                              <1>
451                              <1> ; EQUATES (IBM PC-XT-286 BIOS, 1986, 'POSQEQU.INC')
452                              <1>
453                              <1> ;--------- 8042 COMMANDS --------------------------------------------------------
454                              <1> ENA_KBD             equ   0AEh          ; ENABLE KEYBOARD COMMAND
455                              <1> DIS_KBD             equ   0ADh          ; DISABLE KEYBOARD COMMAND
456                              <1> SHUT_CMD    equ   0FEh          ; CAUSE A SHUTDOWN COMMAND
457                              <1> ;--------- 8042 KEYBOARD INTERFACE AND DIAGNOSTIC CONTROL REGISTERS ------------
458                              <1> STATUS_PORT equ   064h          ; 8042 STATUS PORT
459                              <1> INPT_BUF_FULL       equ   00000010b     ; 1 = +INPUT BUFFER FULL
460                              <1> PORT_A              equ   060h          ; 8042 KEYBOARD SCAN CODE/CONTROL PORT
461                              <1> ;---------- 8042 KEYBOARD RESPONSE ---------------------------------------------
462                              <1> KB_ACK              equ   0FAh          ; ACKNOWLEDGE PROM TRANSMISSION
463                              <1> KB_RESEND   equ   0FEh          ; RESEND REQUEST
464                              <1> KB_OVER_RUN equ   0FFh          ; OVER RUN SCAN CODE
465                              <1> ;---------- KEYBOARD/LED COMMANDS -----------------------------------------------
466                              <1> KB_ENABLE   equ   0F4h          ; KEYBOARD ENABLE
467                              <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:
000010CE FB             <1>       sti                         ; ENABLE INTERRUPTS
                        <1>       ;push  ebp
000010CF 50             <1>       push   eax
000010D0 53             <1>       push   ebx
000010D1 51             <1>       push   ecx
000010D2 52             <1>       push   edx
000010D3 56             <1>       push   esi
000010D4 57             <1>       push   edi
000010D5 1E             <1>       push   ds
000010D6 06             <1>       push   es
000010D7 FC             <1>       cld                         ; FORWARD DIRECTION
000010D8 66B81000       <1>       mov    ax, KDATA
000010DC 8ED8           <1>       mov    ds, ax
000010DE 8EC0           <1>       mov    es, ax
                        <1>       ;
                        <1>       ;-----WAIT FOR KEYBOARD DISABLE COMMAND TO BE ACCEPTED
000010E0 B0AD           <1>       mov    al, DIS_KBD          ; DISABLE THE KEYBOARD COMMAND
000010E2 E8A9050000     <1>       call   SHIP_IT                    ; EXECUTE DISABLE
000010E7 FA             <1>       cli                         ; DISABLE INTERRUPTS
000010E8 B900000100     <1>       mov    ecx, 10000h          ; SET MAXIMUM TIMEOUT
                        <1> KB_INT_01:
000010ED E464           <1>       in     al, STATUS_PORT           ; READ ADAPTER STATUS
000010EF A802           <1>       test   al, INPT_BUF_FULL    ; CHECK INPUT BUFFER FULL STATUS BIT
000010F1 E0FA           <1>       loopnz KB_INT_01            ; WAIT FOR COMMAND TO BE ACCEPTED
                        <1>       ;
```

```
                                <1>       ;----- READ CHARACTER FROM KEYBOARD INTERFACE
000010F3 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
000010F5 FB                     <1>       sti                        ; ENABLE INTERRUPTS AGAIN
000010F6 3CFE                   <1>       cmp    al, KB_RESEND       ; IS THE INPUT A RESEND
000010F8 7411                   <1>         je     short KB_INT_4          ; GO IF RESEND
                                <1>       ;
                                <1>       ;----- CHECK FOR RESPONSE TO A COMMAND TO KEYBOARD
000010FA 3CFA                   <1>       cmp    al, KB_ACK          ; IS THE INPUT AN ACKNOWLEDGE
000010FC 751A                   <1>         jne    short KB_INT_2          ; GO IF NOT
                                <1>       ;
                                <1>       ;----- A COMMAND TO THE KEYBOARD WAS ISSUED
000010FE FA                     <1>       cli                        ; DISABLE INTERRUPTS
000010FF 800D[876F0000]10       <1>       or     byte [KB_FLAG_2], KB_FA ; INDICATE ACK RECEIVED
00001106 E97A020000             <1>         jmp    K26                     ; RETURN IF NOT (ACK RETURNED FOR DATA)
                                <1>       ;
                                <1>       ;----- RESEND THE LAST BYTE
                                <1> KB_INT_4:
0000110B FA                     <1>       cli                        ; DISABLE INTERRUPTS
0000110C 800D[876F0000]20       <1>       or     byte [KB_FLAG_2], KB_FE ; INDICATE RESEND RECEIVED
00001113 E96D020000             <1>         jmp    K26                     ; RETURN IF NOT ACK RETURNED FOR DATA)
                                <1>       ;
                                <1> ;-----      UPDATE MODE INDICATORS IF CHANGE IN STATE
                                <1> KB_INT_2:
00001118 6650                   <1>       push   ax                  ; SAVE DATA IN
0000111A E83D060000             <1>       call   MAKE_LED            ; GO GET MODE INDICATOR DATA BYTE
0000111F 8A1D[876F0000]         <1>       mov    bl, [KB_FLAG_2]     ; GET PREVIOUS BITS
00001125 30C3                   <1>       xor    bl, al              ; SEE IF ANY DIFFERENT
00001127 80E307                 <1>       and    bl, KB_LEDS         ; ISOLATE INDICATOR BITS
0000112A 7405                   <1>       jz     short UP0           ; IF NO CHANGE BYPASS UPDATE
0000112C E8C0050000             <1>       call   SND_LED                   ; GO TURN ON MODE INDICATORS
                                <1> UP0:
00001131 6658                   <1>       pop    ax                  ; RESTORE DATA IN
                                <1> ;--------------------------------------------------------------------------
                                <1> ;     START OF KEY PROCESSING                                        ;
                                <1> ;--------------------------------------------------------------------------
00001133 88C4                   <1>       mov    ah, al              ; SAVE SCAN CODE IN AH ALSO
                                <1>       ;
                                <1>       ;----- TEST FOR OVERRUN SCAN CODE FROM KEYBOARD
00001135 3CFF                   <1>       cmp    al, KB_OVER_RUN           ; IS THIS AN OVERRUN CHAR
00001137 0F843F050000           <1>         je     K62                     ; BUFFER_FULL_BEEP
                                <1>       ;
                                <1> K16:
0000113D 8A3D[886F0000]         <1>       mov    bh, [KB_FLAG_3]           ; LOAD FLAGS FOR TESTING
                                <1>       ;
                                <1>       ;----- TEST TO SEE IF A READ_ID IS IN PROGRESS
00001143 F6C7C0                 <1>       test   bh, RD_ID+LC_AB     ; ARE WE DOING A READ ID?
00001146 7449                   <1>       jz     short NOT_ID        ; CONTINUE IF NOT
00001148 7917                   <1>       jns    short TST_ID_2            ; IS THE RD_ID FLAG ON?
0000114A 3CAB                   <1>       cmp    al, ID_1            ; IS THIS THE 1ST ID CHARACTER?
0000114C 7507                   <1>       jne    short RST_RD_ID
0000114E 800D[886F0000]40       <1>       or     byte [KB_FLAG_3], LC_AB ; INDICATE 1ST ID WAS OK
                                <1> RST_RD_ID:
00001155 8025[886F0000]7F       <1>       and    byte [KB_FLAG_3], ~RD_ID ; RESET THE READ ID FLAG
                                <1>         ;jmp    short ID_EX            ; AND EXIT
0000115C E924020000             <1>       jmp    K26
                                <1>       ;
                                <1> TST_ID_2:
00001161 8025[886F0000]BF       <1>       and    byte [KB_FLAG_3], ~LC_AB ; RESET FLAG
00001168 3C54                   <1>       cmp    al, ID_2A           ; IS THIS THE 2ND ID CHARACTER?
0000116A 7419                   <1>         je   short KX_BIT        ; JUMP IF SO
0000116C 3C41                   <1>       cmp    al, ID_2            ; IS THIS THE 2ND ID CHARACTER?
                                <1>         ;jne short ID_EX         ; LEAVE IF NOT
0000116E 0F8511020000           <1>       jne    K26
                                <1>       ;
                                <1>       ;----- A READ ID SAID THAT IT WAS ENHANCED KEYBOARD
00001174 F6C720                 <1>       test   bh, SET_NUM_LK             ; SHOULD WE SET NUM LOCK?
00001177 740C                   <1>         jz     short KX_BIT           ; EXIT IF NOT
00001179 800D[856F0000]20       <1>       or     byte [KB_FLAG], NUM_STATE ; FORCE NUM LOCK ON
00001180 E86C050000             <1>       call   SND_LED                   ; GO SET THE NUM LOCK INDICATOR
                                <1> KX_BIT:
00001185 800D[886F0000]10       <1>       or     byte [KB_FLAG_3], KBX     ; INDICATE ENHANCED KEYBOARD WAS FOUND
0000118C E9F4010000             <1> ID_EX:      jmp    K26               ; EXIT
                                <1>       ;
                                <1> NOT_ID:
00001191 3CE0                   <1>       cmp    al, MC_E0           ; IS THIS THE GENERAL MARKER CODE?
00001193 750C                   <1>       jne    short TEST_E1
00001195 800D[886F0000]12       <1>       or     byte [KB_FLAG_3], LC_E0+KBX ; SET FLAG BIT, SET KBX, AND
                                <1>       ;jmp   short EXIT          ; THROW AWAY THIS CODE
0000119C E9EB010000             <1>       jmp    K26A
                                <1> TEST_E1:
000011A1 3CE1                   <1>       cmp    al, MC_E1           ; IS THIS THE PAUSE KEY?
000011A3 750C                   <1>       jne    short NOT_HC
000011A5 800D[886F0000]11       <1>       or     byte [KB_FLAG_3], LC_E1+KBX ; SET FLAG BIT, SET KBX, AND
000011AC E9DB010000             <1> EXIT: jmp    K26A                ; THROW AWAY THIS CODE
                                <1>       ;
                                <1> NOT_HC:
000011B1 247F                   <1>       and    al, 07Fh            ; TURN OFF THE BREAK BIT
000011B3 F6C702                 <1>       test   bh, LC_E0           ; LAST CODE THE E0 MARKER CODE
000011B6 7414                   <1>       jz     short NOT_LC_E0           ; JUMP IF NOT
                                <1>       ;
000011B8 BF[726E0000]           <1>       mov    edi, _K6+6          ; IS THIS A SHIFT KEY?
000011BD AE                     <1>       scasb
000011BE 0F84C1010000           <1>         je     K26 ; K16B              ; YES, THROW AWAY & RESET FLAG
```

```
000011C4 AE                <1>       scasb
000011C5 757C              <1>       jne    short K16A          ; NO, CONTINUE KEY PROCESSING
                           <1>       ;jmp   short K16B          ; YES, THROW AWAY & RESET FLAG
000011C7 E9B9010000        <1>       jmp    K26
                           <1>       ;
                           <1> NOT_LC_E0:
000011CC F6C701            <1>       test   bh, LC_E1           ; LAST CODE THE E1 MARKER CODE?
000011CF 7435              <1>       jz     short T_SYS_KEY           ; JUMP IF NOT
000011D1 B904000000        <1>       mov    ecx, 4              ; LENGHT OF SEARCH
000011D6 BF[706E0000]      <1>       mov    edi, _K6+4          ; IS THIS AN ALT, CTL, OR SHIFT?
000011DB F2AE              <1>       repne  scasb               ; CHECK IT
                           <1>       ;je    short EXIT          ; THROW AWAY IF SO
000011DD 0F84A9010000      <1>       je     K26A
                           <1>       ;
000011E3 3C45              <1>       cmp    al, NUM_KEY         ; IS IT THE PAUSE KEY?
                           <1>       ;jne   short K16B          ; NO, THROW AWAY & RESET FLAG
000011E5 0F859A010000      <1>       jne    K26
000011EB F6C480            <1>       test   ah, 80h                   ; YES, IS IT THE BREAK OF THE KEY?
                           <1>       ;jnz   short K16B          ; YES, THROW THIS AWAY, TOO
000011EE 0F8591010000      <1>       jnz    K26
                           <1>         ; 20/02/2015
000011F4 F605[866F0000]08  <1>       test   byte [KB_FLAG_1],HOLD_STATE ;  NO, ARE WE PAUSED ALREADY?
                           <1>       ;jnz   short K16B          ;  YES, THROW AWAY
000011FB 0F8584010000      <1>       jnz    K26
00001201 E9E1020000        <1>       jmp     K39P                    ; NO, THIS IS THE REAL PAUSE STATE
                           <1>       ;
                           <1>       ;----- TEST FOR SYSTEM KEY
                           <1> T_SYS_KEY:
00001206 3C54              <1>       cmp    al, SYS_KEY         ; IS IT THE SYSTEM KEY?
00001208 7539              <1>       jnz    short K16A          ; CONTINUE IF NOT
                           <1>       ;
0000120A F6C480            <1>       test   ah, 80h                   ; CHECK IF THIS A BREAK CODE
0000120D 7524              <1>       jnz    short K16C          ; DO NOT TOUCH SYSTEM INDICATOR IF TRUE
                           <1>       ;
0000120F F605[866F0000]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
00001216 0F8569010000      <1>       jnz     K26
                           <1>       ;
0000121C 800D[866F0000]04  <1>       or     byte [KB_FLAG_1], SYS_SHIFT ; INDICATE SYSTEM KEY DEPRESSED
00001223 B020              <1>       mov    al, EOI                   ; END OF INTERRUPT COMMAND
00001225 E620              <1>       out    20h, al ;out INTA00, al    ; SEND COMMAND TO INTERRUPT CONTROL PORT
                           <1>                                       ; INTERRUPT-RETURN-NO-EOI
00001227 B0AE              <1>       mov    al, ENA_KBD         ; INSURE KEYBOARD IS ENABLED
00001229 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
0000122E E965010000        <1>         jmp     K27A                    ; END PROCESSING
                           <1>       ;
                           <1> ;K16B:       jmp    K26                ; IGNORE SYSTEM KEY
                           <1>       ;
                           <1> K16C:
00001233 8025[866F0000]FB  <1>       and    byte [KB_FLAG_1], ~SYS_SHIFT ; TURN OFF SHIFT KEY HELD DOWN
0000123A B020              <1>       mov    al, EOI                   ; END OF INTERRUPT COMMAND
0000123C 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>       ;
0000123E E94E010000        <1>       jmp     K27                ; IGNORE SYSTEM KEY
                           <1>       ;
                           <1>       ;----- TEST FOR SHIFT KEYS
                           <1> K16A:
00001243 8A1D[856F0000]    <1>       mov    bl, [KB_FLAG]       ; PUT STATE FLAGS IN BL
00001249 BF[6C6E0000]      <1>       mov    edi, _K6            ; SHIFT KEY TABLE offset
0000124E B908000000        <1>       mov    ecx,  K6L           ; LENGTH
00001253 F2AE              <1>       repne  scasb               ; LOOK THROUGH THE TABLE FOR A MATCH
00001255 88E0              <1>       mov    al, ah              ; RECOVER SCAN CODE
00001257 0F8510010000      <1>         jne     K25                     ; IF NO MATCH, THEN SHIFT NOT FOUND
                           <1>       ;
                           <1>       ;------      SHIFT KEY FOUND
                           <1> K17:
0000125D 81EF[6D6E0000]    <1>         sub     edi, _K6+1              ; ADJUST PTR TO SCAN CODE MATCH
00001263 8AA7[746E0000]    <1>             mov     ah, [edi+_K7]           ; GET MASK INTO AH
00001269 B102              <1>       mov    cl, 2               ; SETUP COUNT FOR FLAG SHIFTS
0000126B A880              <1>       test   al, 80h                  ; TEST FOR BREAK KEY
0000126D 0F8596000000      <1>         jnz     K23                     ; JUMP OF BREAK
                           <1>       ;
                           <1>       ;----- SHIFT MAKE FOUND, DETERMINE SET OR TOGGLE
                           <1> K17C:
00001273 80FC10            <1>       cmp    ah, SCROLL_SHIFT
00001276 732B              <1>       jae    short K18           ; IF SCROLL SHIFT OR ABOVE, TOGGLE KEY
                           <1>       ;
                           <1>       ;----- PLAIN SHIFT KEY, SET SHIFT ON
00001278 0825[856F0000]    <1>       or     [KB_FLAG], ah       ; TURN ON SHIFT BIT
0000127E A80C              <1>         test   al, CTL_SHIFT+ALT_SHIFT ; IS IT ALT OR CTRL?
                           <1>       ;jnz   short K17D          ; YES, MORE FLAGS TO SET
00001280 0F84FF000000      <1>       jz     K26                 ; NO, INTERRUPT RETURN
                           <1> K17D:
00001286 F6C702            <1>       test   bh, LC_E0           ; IS THIS ONE OF NEW KEYS?
00001289 740B              <1>       jz     short K17E          ; NO, JUMP
0000128B 0825[886F0000]    <1>       or     [KB_FLAG_3], ah           ; SET BITS FOR RIGHT CTRL, ALT
00001291 E9EF000000        <1>       jmp    K26                 ; INTERRUPT RETURN
                           <1> K17E:
00001296 D2EC              <1>       shr    ah, cl              ; MOVE FLAG BITS TWO POSITIONS
00001298 0825[866F0000]    <1>       or     [KB_FLAG_1], ah           ; SET BITS FOR LEFT CTRL, ALT
0000129E E9E2000000        <1>       jmp    K26
                           <1>       ;
                           <1>       ;----- TOGGLED SHIFT KEY, TEST FOR 1ST MAKE OR NOT
```

```
                                <1> K18:                            ; SHIFT-TOGGLE
000012A3 F6C304                 <1>         test   bl, CTL_SHIFT              ; CHECK CTL SHIFT STATE
                                <1>           ;jz        short K18A               ; JUMP IF NOT CTL STATE
000012A6 0F85C1000000           <1>           jnz     K25                       ; JUMP IF CTL STATE
                                <1> K18A:
000012AC 3C52                   <1>         cmp    al, INS_KEY          ; CHECK FOR INSERT KEY
000012AE 7524                   <1>         jne    short K22            ; JUMP IF NOT INSERT KEY
000012B0 F6C308                 <1>         test   bl, ALT_SHIFT              ; CHECK FOR ALTERNATE SHIFT
                                <1>                ;jz    short K18B           ; JUMP IF NOT ALTERNATE SHIFT
000012B3 0F85B4000000           <1>           jnz     K25                       ; JUMP IF ALTERNATE SHIFT
                                <1> K18B:
000012B9 F6C702                 <1>         test   bh, LC_E0 ;20/02/2015      ; IS THIS NEW INSERT KEY?
000012BC 7516                   <1>         jnz    short K22            ; YES, THIS ONE'S NEVER A '0'
                                <1> K19:
000012BE F6C320                 <1>         test   bl, NUM_STATE              ; CHECK FOR BASE STATE
000012C1 750C                   <1>         jnz    short K21            ; JUMP IF NUM LOCK IS ON
000012C3 F6C303                 <1>         test   bl, LEFT_SHIFT+RIGHT_SHIFT ; TEST FOR SHIFT STATE
000012C6 740C                   <1>         jz     short K22            ; JUMP IF BASE STATE
                                <1> K20:                            ; NUMERIC ZERO, NOT INSERT KEY
000012C8 88C4                   <1>         mov    ah, al               ; PUT SCAN CODE BACK IN AH
000012CA E99E000000             <1>           jmp     K25                       ; NUMERAL '0', STNDRD. PROCESSING
                                <1> K21:                            ; MIGHT BE NUMERIC
000012CF F6C303                 <1>         test   bl, LEFT_SHIFT+RIGHT_SHIFT
000012D2 74F4                   <1>         jz     short K20            ; IS NUMERIC, STD. PROC.
                                <1>         ;
                                <1> K22:                            ; SHIFT TOGGLE KEY HIT; PROCESS IT
000012D4 8425[866F0000]         <1>         test   ah, [KB_FLAG_1]      ; IS KEY ALREADY DEPRESSED
000012DA 0F85A5000000           <1>           jnz     K26                       ; JUMP IF KEY ALREADY DEPRESSED
                                <1> K22A:
000012E0 0825[866F0000]         <1>           or      [KB_FLAG_1], ah ; INDICATE THAT THE KEY IS DEPRESSED
000012E6 3025[856F0000]         <1>         xor    [KB_FLAG], ah        ; TOGGLE THE SHIFT STATE
                                <1>         ;
                                <1>         ;----- TOGGLE LED IF CAPS, NUM  OR SCROLL KEY DEPRESSED
000012EC F6C470                 <1>         test   ah, CAPS_SHIFT+NUM_SHIFT+SCROLL_SHIFT ; SHIFT TOGGLE?
000012EF 7409                   <1>         jz     short K22B           ; GO IF NOT
                                <1>         ;
000012F1 6650                   <1>         push   ax                   ; SAVE SCAN CODE AND SHIFT MASK
000012F3 E8F9030000             <1>         call   SND_LED                    ; GO TURN MODE INDICATORS ON
000012F8 6658                   <1>         pop    ax                   ; RESTORE SCAN CODE
                                <1> K22B:
000012FA 3C52                   <1>         cmp    al, INS_KEY          ; TEST FOR 1ST MAKE OF INSERT KEY
000012FC 0F8583000000           <1>           jne     K26                       ; JUMP IF NOT INSERT KEY
00001302 88C4                   <1>         mov    ah, al               ; SCAN CODE IN BOTH HALVES OF AX
00001304 E999000000             <1>           jmp     K28                       ; FLAGS UPDATED, PROC. FOR BUFFER
                                <1>         ;
                                <1>         ;----- BREAK SHIFT FOUND
                                <1> K23:                            ; BREAK-SHIFT-FOUND
00001309 80FC10                 <1>         cmp    ah, SCROLL_SHIFT     ; IS THIS A TOGGLE KEY
0000130C F6D4                   <1>         not    ah                   ; INVERT MASK
0000130E 7355                   <1>         jae    short K24            ; YES, HANDLE BREAK TOGGLE
00001310 2025[856F0000]         <1>         and    [KB_FLAG], ah        ; TURN OFF SHIFT BIT
00001316 80FCFB                 <1>         cmp    ah, ~CTL_SHIFT              ; IS THIS ALT OR CTL?
00001319 7730                   <1>         ja     short K23D           ; NO, ALL DONE
                                <1>         ;
0000131B F6C702                 <1>         test   bh, LC_E0            ; 2ND ALT OR CTL?
0000131E 7408                   <1>         jz     short K23A           ; NO, HANSLE NORMALLY
00001320 2025[886F0000]         <1>         and    [KB_FLAG_3], ah              ; RESET BIT FOR RIGHT ALT OR CTL
00001326 EB08                   <1>         jmp    short K23B           ; CONTINUE
                                <1> K23A:
00001328 D2FC                   <1>         sar    ah, cl               ; MOVE THE MASK BIT TWO POSITIONS
0000132A 2025[866F0000]         <1>         and    [KB_FLAG_1], ah              ; RESET BIT FOR LEFT ALT AND CTL
                                <1> K23B:
00001330 88C4                   <1>         mov    ah, al               ; SAVE SCAN CODE
00001332 A0[886F0000]           <1>         mov    al, [KB_FLAG_3]              ; GET RIGHT ALT & CTRL FLAGS
00001337 D2E8                   <1>         shr    al, cl               ; MOVE TO BITS 1 & 0
00001339 0A05[866F0000]         <1>         or     al, [KB_FLAG_1]              ; PUT IN LEFT ALŞT & CTL FLAGS
0000133F D2E0                   <1>         shl    al, cl               ; MOVE BACK TO BITS 3 & 2
00001341 240C                   <1>         and    al, ALT_SHIFT+CTL_SHIFT ; FILTER OUT OTHER GARBAGE
00001343 0805[856F0000]         <1>         or     [KB_FLAG], al        ; PUT RESULT IN THE REAL FLAGS
00001349 88E0                   <1>         mov    al, ah
                                <1> K23D:
0000134B 3CB8                   <1>         cmp    al, ALT_KEY+80h              ; IS THIS ALTERNATE SHIFT RELEASE
0000134D 7536                   <1>         jne    short K26            ; INTERRUPT RETURN
                                <1>         ;
                                <1>         ;----- ALTERNATE SHIFT KEY RELEASED, GET THE VALUE INTO BUFFER
0000134F A0[896F0000]           <1>         mov    al, [ALT_INPUT]
00001354 B400                   <1>         mov    ah, 0                ; SCAN CODE OF 0
00001356 8825[896F0000]         <1>         mov    [ALT_INPUT], ah      ; ZERO OUT THE FIELD
0000135C 3C00                   <1>         cmp    al, 0                ; WAS THE INPUT = 0?
0000135E 7425                   <1>         je     short K26            ; INTERRUPT_RETURN
                                <1>           ; 29/01/2016
                                <1>         ;jmp     K61                      ; IT WASN'T, SO PUT IN BUFFER
00001360 E9D0020000             <1>         jmp    _K60
                                <1>         ;
                                <1> K24:                            ; BREAK-TOGGLE
00001365 2025[866F0000]         <1>         and    [KB_FLAG_1], ah      ; INDICATE NO LONGER DEPRESSED
0000136B EB18                   <1>         jmp    short K26            ; INTERRUPT_RETURN
                                <1>         ;
                                <1>         ;----- TEST FOR HOLD STATE
                                <1>                                     ; AL, AH = SCAN CODE
                                <1> K25:                            ; NO-SHIFT-FOUND
0000136D 3C80                   <1>         cmp    al, 80h                   ; TEST FOR BREAK KEY
0000136F 7314                   <1>         jae    short K26            ; NOTHING FOR BREAK CHARS FROM HERE ON
00001371 F605[866F0000]08       <1>         test   byte [KB_FLAG_1], HOLD_STATE ; ARE WE IN HOLD STATE
00001378 7428                   <1>         jz     short K28            ; BRANCH AROUND TEST IF NOT
0000137A 3C45                   <1>         cmp    al, NUM_KEY
0000137C 7407                   <1>         je     short K26            ; CAN'T END HOLD ON NUM_LOCK
0000137E 8025[866F0000]F7       <1>         and    byte [KB_FLAG_1], ~HOLD_STATE ; TURN OFF THE HOLD STATE BIT
                                <1>         ;
                                <1> K26:
00001385 8025[886F0000]FC       <1>         and    byte [KB_FLAG_3], ~(LC_E0+LC_E1) ; RESET LAST CHAR H.C. FLAG
                                <1> K26A:                           ; INTERRUPT-RETURN
0000138C FA                     <1>         cli                         ; TURN OFF INTERRUPTS
0000138D B020                   <1>         mov    al, EOI                   ; END OF INTERRUPT COMMAND
0000138F E620                   <1>         out    20h, al      ;out INTA00, al     ; SEND COMMAND TO INTERRUPT CONTROL PORT
```

```
                        <1> K27:                            ; INTERRUPT-RETURN-NO-EOI
00001391 B0AE           <1>         mov    al, ENA_KBD      ; INSURE KEYBOARD IS ENABLED
00001393 E8F8020000     <1>         call   SHIP_IT                  ; EXECUTE ENABLE
                        <1> K27A:
00001398 FA             <1>         cli                     ; DISABLE INTERRUPTS
                        <1>         ;;mov  byte [intflg], 0 ; 07/01/2017 ;; 15/01/2017
00001399 07             <1>         pop    es               ; RESTORE REGISTERS
0000139A 1F             <1>         pop    ds
0000139B 5F             <1>         pop    edi
0000139C 5E             <1>         pop    esi
0000139D 5A             <1>         pop    edx
0000139E 59             <1>         pop    ecx
0000139F 5B             <1>         pop    ebx
000013A0 58             <1>         pop    eax
                        <1>         ;pop   ebp
000013A1 CF             <1>         iretd                   ; RETURN
                        <1>
                        <1>         ;----- NOT IN HOLD STATE
                        <1> K28:                            ; NO-HOLD-STATE
000013A2 3C58           <1>         cmp    al, 88           ; TEST FOR OUT-OF-RANGE SCAN CODES
000013A4 77DF           <1>         ja     short K26        ; IGNORE IF OUT-OF-RANGE
                        <1>         ;
000013A6 F6C308         <1>         test   bl, ALT_SHIFT            ; ARE WE IN ALTERNATE SHIFT
                        <1>           ;jz  short K28A       ; IF NOT ALTERNATE
000013A9 0F84F1000000   <1>           jz     K38
                        <1>         ;
000013AF F6C710         <1>         test   bh, KBX                  ; IS THIS THE ENCHANCED KEYBOARD?
000013B2 740D           <1>         jz     short K29        ; NO, ALT STATE IS REAL
                        <1>          ;28/02/2015
000013B4 F605[866F0000]04 <1>       test   byte [KB_FLAG_1], SYS_SHIFT ; YES, IS SYSREQ KEY DOWN?
                        <1>         ;jz    short K29        ;  NO, ALT STATE IS REAL
000013BB 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
000013C1 F6C304         <1>         test   bl, CTL_SHIFT            ; ARE WE IN CONTROL SHIFT ALSO?
000013C4 740B           <1>         jz     short K31        ; NO_RESET
000013C6 3C53           <1>         cmp    al, DEL_KEY      ; CTL-ALT STATE, TEST FOR DELETE KEY
000013C8 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.
000013CA B0FE           <1>         mov    al, SHUT_CMD     ; SHUTDOWN COMMAND
000013CC E664           <1>         out    STATUS_PORT, al          ; SEND TO KEYBOARD CONTROL PORT
                        <1> khere:
000013CE F4             <1>         hlt                     ; WAIT FOR 80286 RESET
000013CF EBFD           <1>         jmp    short khere      ; INSURE HALT
                        <1>
                        <1>         ;
                        <1>         ;----- IN ALTERNATE SHIFT, RESET NOT FOUND
                        <1> K31:                            ; NO-RESET
000013D1 3C39           <1>         cmp    al, 57           ; TEST FOR SPACE KEY
000013D3 7507           <1>         jne    short K311       ; NOT THERE
000013D5 B020           <1>         mov    al, ' '                  ; SET SPACE CHAR
000013D7 E948020000     <1>           jmp    K57                    ; BUFFER_FILL
                        <1> K311:
000013DC 3C0F           <1>         cmp    al, 15           ; TEST FOR TAB KEY
000013DE 7509           <1>         jne    short K312       ; NOT THERE
000013E0 66B800A5       <1>         mov    ax, 0A500h       ; SET SPECIAL CODE FOR ALT-TAB
000013E4 E93B020000     <1>           jmp    K57                    ; BUFFER_FILL
                        <1> K312:
000013E9 3C4A           <1>         cmp    al, 74           ; TEST FOR KEY PAD -
000013EB 0F84A2000000   <1>           je     K37B                   ; GO PROCESS
000013F1 3C4E           <1>         cmp    al, 78           ; TEST FOR KEY PAD +
000013F3 0F849A000000   <1>           je     K37B                   ; GO PROCESS
                        <1>         ;
                        <1>         ;----- LOOK FOR KEY PAD ENTRY
                        <1> K32:                            ; ALT-KEY-PAD
000013F9 BF[486E0000]   <1>         mov    edi, K30         ; ALT-INPUT-TABLE offset
000013FE B90A000000     <1>         mov    ecx, 10                  ; LOOK FOR ENTRY USING KEYPAD
00001403 F2AE           <1>         repne  scasb            ; LOOK FOR MATCH
00001405 7525           <1>         jne    short K33        ; NO_ALT_KEYPAD
00001407 F6C702         <1>         test   bh, LC_E0        ; IS THIS ONE OF THE NEW KEYS?
0000140A 0F858A000000   <1>           jnz    K37C                   ; YES, JUMP, NOT NUMPAD KEY
00001410 81EF[496E0000] <1>         sub    edi, K30+1       ; DI NOW HAS ENTRY VALUE
00001416 A0[896F0000]   <1>         mov    al, [ALT_INPUT]  ; GET THE CURRENT BYTE
0000141B B40A           <1>         mov    ah, 10           ; MULTIPLY BY 10
0000141D F6E4           <1>         mul    ah
0000141F 6601F8         <1>         add    ax, di           ; ADD IN THE LATEST ENTRY
00001422 A2[896F0000]   <1>         mov    [ALT_INPUT], al  ; STORE IT AWAY
                        <1> ;K32A:
00001427 E959FFFFFF     <1>           jmp    K26                    ; THROW AWAY THAT KEYSTROKE
                        <1>         ;
                        <1>         ;----- LOOK FOR SUPERSHIFT ENTRY
                        <1> K33:                            ; NO-ALT-KEYPAD
0000142C C605[896F0000]00 <1>         mov    byte [ALT_INPUT], 0      ; ZERO ANY PREVIOUS ENTRY INTO INPUT
00001433 B91A000000     <1>         mov    ecx, 26                  ; (DI),(ES) ALREADY POINTING
00001438 F2AE           <1>         repne  scasb            ; LOOK FOR MATCH IN ALPHABET
0000143A 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
0000143C 3C02           <1>         cmp    al, 2            ; KEY WITH '1' ON IT
0000143E 7253           <1>         jb     short K37B       ; MUST BE ESCAPE
00001440 3C0D           <1>         cmp    al, 13           ; IS IT IN THE REGION
00001442 7705           <1>         ja     short K35        ; NO, ALT SOMETHING ELSE
00001444 80C476         <1>         add    ah, 118                  ; CONVERT PSEUDO SCAN CODE TO RANGE
00001447 EB43           <1>         jmp    short K37A       ; GO FILL THE BUFFER
                        <1>         ;
                        <1>         ;----- TRANSLATE ALTERNATE SHIFT PSEUDO SCAN CODES
```

```
 993                                    <1> K35:                            ; ALT-FUNCTION
 994 00001449 3C57                      <1>         cmp    al, F11_M            ; IS IT F11?
 995 0000144B 7209                      <1>         jb     short K35A ; 20/02/2015   ; NO, BRANCH
 996 0000144D 3C58                      <1>         cmp    al, F12_M            ; IS IT F12?
 997 0000144F 7705                      <1>         ja     short K35A ; 20/02/2015   ; NO, BRANCH
 998 00001451 80C434                    <1>         add    ah, 52               ; CONVERT TO PSEUDO SCAN CODE
 999 00001454 EB36                      <1>         jmp    short K37A           ; GO FILL THE BUFFER
1000                                    <1> K35A:
1001 00001456 F6C702                    <1>         test   bh, LC_E0            ; DO WE HAVE ONE OF THE NEW KEYS?
1002 00001459 7422                      <1>         jz     short K37            ; NO, JUMP
1003 0000145B 3C1C                      <1>         cmp    al, 28               ; TEST FOR KEYPAD ENTER
1004 0000145D 7509                      <1>           jne     short K35B              ; NOT THERE
1005 0000145F 66B800A6                  <1>         mov    ax, 0A600h           ; SPECIAL CODE
1006 00001463 E9BC010000                <1>         jmp    K57                  ; BUFFER FILL
1007                                    <1> K35B:
1008 00001468 3C53                      <1>         cmp    al, 83               ; TEST FOR DELETE KEY
1009 0000146A 742E                      <1>         je     short K37C           ; HANDLE WITH OTHER EDIT KEYS
1010 0000146C 3C35                      <1>         cmp    al, 53               ; TEST FOR KEYPAD /
1011                                    <1>         ;jne   short K32A           ; NOT THERE, NO OTHER E0 SPECIALS
1012 0000146E 0F8511FFFFFF              <1>           jne     K26
1013 00001474 66B800A4                  <1>         mov    ax, 0A400h           ; SPECIAL CODE
1014 00001478 E9A7010000                <1>         jmp    K57                  ; BUFFER FILL
1015                                    <1> K37:
1016 0000147D 3C3B                      <1>         cmp    al, 59               ; TEST FOR FUNCTION KEYS (F1)
1017 0000147F 7212                      <1>           jb      short K37B             ; NO FN, HANDLE W/OTHER EXTENDED
1018 00001481 3C44                      <1>         cmp    al, 68               ; IN KEYPAD REGION?
1019                                    <1>           ;ja  short K32A           ; IF SO, IGNORE
1020 00001483 0F87FCFEFFFF              <1>           ja      K26
1021 00001489 80C42D                    <1>         add    ah, 45               ; CONVERT TO PSEUDO SCAN CODE
1022                                    <1> K37A:
1023 0000148C B000                      <1>         mov    al, 0                ; ASCII CODE OF ZERO
1024 0000148E E991010000                <1>           jmp     K57                     ; PUT IT IN THE BUFFER
1025                                    <1> K37B:
1026 00001493 B0F0                      <1>         mov    al, 0F0h             ; USE SPECIAL ASCII CODE
1027 00001495 E98A010000                <1>         jmp     K57                     ; PUT IT IN THE BUFFER
1028                                    <1> K37C:
1029 0000149A 0450                      <1>         add    al, 80               ; CONVERT SCAN CODE (EDIT KEYS)
1030 0000149C 88C4                      <1>         mov    ah, al               ; (SCAN CODE NOT IN AH FOR INSERT)
1031 0000149E 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 000014A0 F6C304                    <1>         test   bl, CTL_SHIFT              ; ARE WE IN CONTROL SHIFT?
1037                                    <1>         ;jnz   short K38A           ; YES, START PROCESSING
1038 000014A3 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 000014A9 3C46                      <1>         cmp    al, SCROLL_KEY             ; TEST FOR BREAK
1044 000014AB 7531                      <1>         jne    short K39            ; JUMP, NO-BREAK
1045 000014AD F6C710                    <1>         test   bh, KBX                    ; IS THIS THE ENHANCED KEYBOARD?
1046 000014B0 7405                      <1>         jz     short K38B           ; NO, BREAK IS VALID
1047 000014B2 F6C702                    <1>         test   bh, LC_E0            ; YES, WAS LAST CODE AN E0?
1048 000014B5 7427                      <1>         jz     short K39            ; NO-BREAK, TEST FOR PAUSE
1049                                    <1> K38B:
1050 000014B7 8B1D[926F0000]            <1>         mov    ebx, [BUFFER_HEAD]  ; RESET BUFFER TO EMPTY
1051 000014BD 891D[966F0000]            <1>         mov    [BUFFER_TAIL], ebx
1052 000014C3 C605[846F0000]80          <1>         mov    byte [BIOS_BREAK], 80h  ; TURN ON BIOS_BREAK BIT
1053                                    <1>         ;
1054                                    <1>         ;----- ENABLE KEYBOARD
1055 000014CA B0AE                      <1>         mov    al, ENA_KBD          ; ENABLE KEYBOARD
1056 000014CC 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 000014D1 E8FA600000                <1>         call   ctrlbrk ; control+break subroutine
1062                                    <1>         ;
1063 000014D6 6629C0                    <1>         sub    ax, ax               ; PUT OUT DUMMY CHARACTER
1064 000014D9 E946010000                <1>           jmp     K57                     ; BUFFER_FILL
1065                                    <1>         ;
1066                                    <1>         ;----- TEST FOR PAUSE
1067                                    <1> K39:                            ; NO_BREAK
1068 000014DE F6C710                    <1>         test   bh, KBX                    ; IS THIS THE ENHANCED KEYBOARD?
1069 000014E1 7537                      <1>         jnz    short K41            ; YES, THEN THIS CAN'T BE PAUSE
1070 000014E3 3C45                      <1>         cmp    al, NUM_KEY          ; LOOK FOR PAUSE KEY
1071 000014E5 7533                      <1>         jne    short K41            ; NO-PAUSE
1072                                    <1> K39P:
1073 000014E7 800D[866F0000]08          <1>         or     byte [KB_FLAG_1], HOLD_STATE ; TURN ON THE HOLD FLAG
1074                                    <1>         ;
1075                                    <1>         ;----- ENABLE KEYBOARD
1076 000014EE B0AE                      <1>         mov    al, ENA_KBD          ; ENABLE KEYBOARD
1077 000014F0 E89B010000                <1>         call   SHIP_IT                    ; EXECUTE ENABLE
1078                                    <1> K39A:
1079 000014F5 B020                      <1>         mov    al, EOI                    ; END OF INTERRUPT TO CONTROL PORT
1080 000014F7 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 000014F9 803D[BA6F0000]07          <1>           cmp     byte [CRT_MODE], 7      ; IS THIS BLACK AND WHITE CARD
1084 00001500 740A                      <1>           je      short K40                  ; YES, NOTHING TO DO
1085 00001502 66BAD803                  <1>         mov    dx, 03D8h            ; PORT FOR COLOR CARD
1086 00001506 A0[BB6F0000]              <1>           mov     al, [CRT_MODE_SET]     ; GET THE VALUE OF THE CURRENT MODE
1087 0000150B EE                        <1>         out    dx, al               ; SET THE CRT MODE, SO THAT CRT IS ON
1088                                    <1>         ;
1089                                    <1> K40:                            ; PAUSE-LOOP
1090 0000150C F605[866F0000]08          <1>           test    byte [KB_FLAG_1], HOLD_STATE ; CHECK HOLD STATE FLAG
1091 00001513 75F7                      <1>         jnz    short K40            ; LOOP UNTIL FLAG TURNED OFF
1092                                    <1>         ;
1093 00001515 E977FEFFFF                <1>           jmp     K27                     ; INTERRUPT_RETURN_NO_EOI
1094                                    <1>           ;
1095                                    <1>         ;----- TEST SPECIAL CASE KEY 55
1096                                    <1> K41:                            ; NO-PAUSE
1097 0000151A 3C37                      <1>         cmp    al, 55               ; TEST FOR */PRTSC KEY
```

```
1098 0000151C 7513              <1>       jne    short K42           ; NOT-KEY-55
1099 0000151E F6C710            <1>       test   bh, KBX                    ; IS THIS THE ENHANCED KEYBOARD?
1100 00001521 7405              <1>       jz     short K41A          ; NO, CTL-PRTSC IS VALID
1101 00001523 F6C702            <1>       test   bh, LC_E0           ; YES, WAS LAST CODE AN E0?
1102 00001526 7421              <1>       jz     short K42B          ; NO, TRANSLATE TO A FUNCTION
1103                            <1> K41A:
1104 00001528 66B80072          <1>       mov    ax, 114*256         ; START/STOP PRINTING SWITCH
1105 0000152C E9F3000000        <1>         jmp     K57                      ; BUFFER_FILL
1106                            <1>       ;
1107                            <1>       ;----- SET UP TO TRANSLATE CONTROL SHIFT
1108                            <1> K42:                              ; NOT-KEY-55
1109 00001531 3C0F              <1>       cmp    al, 15              ; IS IT THE TAB KEY?
1110 00001533 7414              <1>       je     short K42B          ; YES, XLATE TO FUNCTION CODE
1111 00001535 3C35              <1>       cmp    al, 53              ; IS IT THE / KEY?
1112 00001537 750E              <1>       jne    short K42A          ; NO, NO MORE SPECIAL CASES
1113 00001539 F6C702            <1>       test   bh, LC_E0           ; YES, IS IT FROM THE KEY PAD?
1114 0000153C 7409              <1>       jz     short K42A          ; NO, JUST TRANSLATE
1115 0000153E 66B80095          <1>       mov    ax, 9500h           ; YES, SPECIAL CODE FOR THIS ONE
1116 00001542 E9DD000000        <1>       jmp    K57                 ; BUFFER FILL
1117                            <1> K42A:
1118                            <1>       ;;mov  ebx, _K8            ; SET UP TO TRANSLATE CTL
1119 00001547 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 00001549 BB[7C6E0000]      <1>       mov    ebx, _K8            ; SET UP TO TRANSLATE CTL
1125 0000154E 0F82AE000000      <1>       jb     K56 ;; 20/02/2015
1126 00001554 E9B9000000        <1>       jmp    K64
1127                            <1>         ;
1128                            <1>       ;----- NOT IN CONTROL SHIFT
1129                            <1> K44:                              ; NOT-CTL-SHIFT
1130 00001559 3C37              <1>       cmp    al, 55              ; PRINT SCREEN KEY?
1131 0000155B 7528              <1>       jne    short K45           ; NOT PRINT SCREEN
1132 0000155D F6C710            <1>       test   bh, KBX                    ; IS THIS ENHANCED KEYBOARD?
1133 00001560 7407              <1>       jz     short K44A          ; NO, TEST FOR SHIFT STATE
1134 00001562 F6C702            <1>       test   bh, LC_E0           ; YES, LAST CODE A MARKER?
1135 00001565 7507              <1>       jnz    short K44B          ; YES, IS PRINT SCREEN
1136 00001567 EB41              <1>       jmp    short K45C          ; NO, TRANSLATE TO '*' CHARACTER
1137                            <1> K44A:
1138 00001569 F6C303            <1>       test   bl, LEFT_SHIFT+RIGHT_SHIFT ; NOT 101 KBD, SHIFT KEY DOWN?
1139 0000156C 743C              <1>       jz     short K45C          ; NO, TRANSLATE TO '*' CHARACTER
1140                            <1>       ;
1141                            <1>       ;----- ISSUE INTERRUPT TO INDICATE PRINT SCREEN FUNCTION
1142                            <1> K44B:
1143 0000156E B0AE              <1>       mov    al, ENA_KBD         ; INSURE KEYBOARD IS ENABLED
1144 00001570 E81B010000        <1>       call   SHIP_IT                    ; EXECUTE ENABLE
1145 00001575 B020              <1>       mov    al, EOI                    ; END OF CURRENT INTERRUPT
1146 00001577 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 00001579 8025[886F0000]FC  <1>         and     byte [KB_FLAG_3], ~(LC_E0+LC_E1) ; ZERO OUT THESE FLAGS
1152 00001580 E90CFEFFFF        <1>         jmp     K27                      ; GO BACK WITHOUT EOI OCCURRING
1153                            <1>       ;
1154                            <1>       ;----- HANDLE IN-CORE KEYS
1155                            <1> K45:                              ; NOT-PRINT-SCREEN
1156 00001585 3C3A              <1>       cmp    al, 58              ; TEST FOR IN-CORE AREA
1157 00001587 7734              <1>       ja     short K46           ; JUMP IF NOT
1158 00001589 3C35              <1>       cmp    al, 53              ; IS THIS THE '/' KEY?
1159 0000158B 7505              <1>       jne    short K45A          ; NO, JUMP
1160 0000158D F6C702            <1>       test   bh, LC_E0           ; WAS THE LAST CODE THE MARKER?
1161 00001590 7518              <1>       jnz    short K45C          ; YES, TRANSLATE TO CHARACTER
1162                            <1> K45A:
1163 00001592 B91A000000        <1>       mov    ecx, 26                    ; LENGHT OF SEARCH
1164 00001597 BF[526E0000]      <1>       mov    edi, K30+10         ; POINT TO TABLE OF A-Z CHARS
1165 0000159C F2AE              <1>       repne  scasb               ; IS THIS A LETTER KEY?
1166                            <1>              ; 20/02/2015
1167 0000159E 7505              <1>       jne    short K45B             ; NO, SYMBOL KEY
1168                            <1>       ;
1169 000015A0 F6C340            <1>       test   bl, CAPS_STATE             ; ARE WE IN CAPS LOCK?
1170 000015A3 750C              <1>       jnz    short K45D          ; TEST FOR SURE
1171                            <1> K45B:
1172 000015A5 F6C303            <1>       test   bl, LEFT_SHIFT+RIGHT_SHIFT ; ARE WE IN SHIFT STATE?
1173 000015A8 750C              <1>       jnz    short K45E          ; YES, UPPERCASE
1174                            <1>                                  ; NO, LOWERCASE
1175                            <1> K45C:
1176 000015AA BB[D46E0000]      <1>       mov    ebx, K10            ; TRANSLATE TO LOWERCASE LETTERS
1177 000015AF EB51              <1>       jmp    short K56
1178                            <1> K45D:                             ; ALMOST-CAPS-STATE
1179 000015B1 F6C303            <1>       test   bl, LEFT_SHIFT+RIGHT_SHIFT ; CL ON. IS SHIFT ON, TOO?
1180 000015B4 75F4              <1>       jnz    short K45C          ; SHIFTED TEMP OUT OF CAPS STATE
1181                            <1> K45E:
1182 000015B6 BB[2C6F0000]      <1>       mov    ebx, K11            ; TRANSLATE TO UPPER CASE LETTERS
1183 000015BB EB45              <1> K45F: jmp    short K56
1184                            <1>       ;
1185                            <1>       ;----- TEST FOR KEYS F1 - F10
1186                            <1> K46:                              ; NOT IN-CORE AREA
1187 000015BD 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 000015BF 7635              <1>       jna    short K53
1191                            <1>       ;
1192                            <1>       ;----- HANDLE THE NUMERIC PAD KEYS
1193                            <1> K47:                              ; NOT F1 - F10
1194 000015C1 3C53              <1>       cmp    al, 83              ; TEST NUMPAD KEYS
1195 000015C3 772D              <1>       ja     short K52           ; JUMP IF NOT
1196                            <1>       ;
1197                            <1>       ;----- KEYPAD KEYS, MUST TEST NUM LOCK FOR DETERMINATION
1198                            <1> K48:
1199 000015C5 3C4A              <1>       cmp    al , 74                    ; SPECIAL CASE FOR MINUS
1200 000015C7 74ED              <1>       je     short K45E          ; GO TRANSLATE
1201 000015C9 3C4E              <1>       cmp    al , 78                    ; SPECIAL CASE FOR PLUS
1202 000015CB 74E9              <1>       je     short K45E          ; GO TRANSLATE
```

```
1203 000015CD F6C702             <1>       test   bh, LC_E0           ; IS THIS ONE OFTHE NEW KEYS?
1204 000015D0 750A               <1>       jnz    short K49           ; YES, TRANSLATE TO BASE STATE
1205                             <1>       ;
1206 000015D2 F6C320             <1>       test   bl, NUM_STATE       ; ARE WE IN NUM LOCK
1207 000015D5 7514               <1>       jnz    short K50           ; TEST FOR SURE
1208 000015D7 F6C303             <1>       test   bl, LEFT_SHIFT+RIGHT_SHIFT ; ARE WE IN SHIFT STATE?
1209                             <1>       ;jnz   short K51           ; IF SHIFTED, REALLY NUM STATE
1210 000015DA 75DA               <1>       jnz    short K45E
1211                             <1>       ;
1212                             <1>       ;----- BASE CASE FOR KEYPAD
1213                             <1> K49:
1214 000015DC 3C4C               <1>       cmp    al, 76              ; SPECIAL CASE FOR BASE STATE 5
1215 000015DE 7504               <1>       jne    short K49A          ; CONTINUE IF NOT KEYPAD 5
1216 000015E0 B0F0               <1>       mov    al, 0F0h            ; SPECIAL ASCII CODE
1217 000015E2 EB40               <1>       jmp    short K57           ; BUFFER FILL
1218                             <1> K49A:
1219 000015E4 BB[D46E0000]       <1>       mov    ebx, K10            ; BASE CASE TABLE
1220 000015E9 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 000015EB F6C303             <1>         test    bl, LEFT_SHIFT+RIGHT_SHIFT
1225 000015EE 75EC               <1>       jnz    short K49           ; SHIFTED TEMP OUT OF NUM STATE
1226 000015F0 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 000015F2 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 000015F4 74AF               <1>       je     short K45B
1234                             <1>       ;
1235                             <1>       ;----- MUST BE F11 OR F12
1236                             <1> K53:                             ; F1 - F10 COME HERE, TOO
1237 000015F6 F6C303             <1>       test   bl, LEFT_SHIFT+RIGHT_SHIFT ; TEST SHIFT STATE
1238 000015F9 74E1               <1>       jz     short K49           ; JUMP, LOWER CASE PSEUDO SC'S
1239                             <1>              ; 20/02/2015
1240 000015FB BB[2C6F0000]       <1>       mov    ebx, K11            ; UPPER CASE PSEUDO SCAN CODES
1241 00001600 EB10               <1>       jmp    short K64           ; TRANSLATE SCAN
1242                             <1>       ;
1243                             <1>       ;----- TRANSLATE THE CHARACTER
1244                             <1> K56:                             ; TRANSLATE-CHAR
1245 00001602 FEC8               <1>       dec    al                  ; CONVERT ORIGIN
1246 00001604 D7                 <1>       xlat                       ; CONVERT THE SCAN CODE TO ASCII
1247 00001605 F605[886F0000]02   <1>       test   byte [KB_FLAG_3], LC_E0   ; IS THIS A NEW KEY?
1248 0000160C 7416               <1>       jz     short K57           ; NO, GO FILL BUFFER
1249 0000160E B4E0               <1>       mov    ah, MC_E0           ; YES, PUT SPECIAL MARKER IN AH
1250 00001610 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 00001612 FEC8               <1>       dec    al                  ; CONVERT ORIGIN
1255 00001614 D7                 <1>              xlat                        ; CTL TABLE SCAN
1256 00001615 88C4               <1>       mov    ah, al              ; PUT VALUE INTO AH
1257 00001617 B000               <1>       mov    al, 0               ; ZERO ASCII CODE
1258 00001619 F605[886F0000]02   <1>       test   byte [KB_FLAG_3], LC_E0   ; IS THIS A NEW KEY?
1259 00001620 7402               <1>       jz     short K57           ; NO, GO FILL BUFFER
1260 00001622 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 00001624 3CFF               <1>       cmp    al, -1              ; IS THIS AN IGNORE CHAR
1265                             <1>         ;je  short K59           ; YES, DO NOTHING WITH IT
1266 00001626 0F8459FDFFFF       <1>       je       K26               ; YES, DO NOTHING WITH IT
1267 0000162C 80FCFF             <1>       cmp    ah, -1              ; LOOK FOR -1 PSEUDO SCAN
1268                             <1>         ;jne short K61           ; NEAR_INTERRUPT_RETURN
1269 0000162F 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 00001635 80FC68             <1>       cmp    ah, 68h      ; ALT + F1 key
1275 00001638 721F               <1>       jb     short K61
1276 0000163A 80FC6F             <1>       cmp    ah, 6Fh ; ALT + F8 key
1277 0000163D 771A               <1>       ja     short K61
1278                             <1>       ;
1279 0000163F 8A1D[BE810100]     <1>       mov    bl, [ACTIVE_PAGE]
1280 00001645 80C368             <1>       add    bl, 68h
1281 00001648 38E3               <1>       cmp    bl, ah
1282 0000164A 740D               <1>       je     short K61
1283 0000164C 6650               <1>       push   ax
1284 0000164E 88E0               <1>       mov    al, ah
1285 00001650 2C68               <1>       sub    al, 68h
1286 00001652 E858090000         <1>       call   set_active_page
1287 00001657 6658               <1>       pop    ax
1288                             <1> K61:                             ; NOT-CAPS-STATE
1289 00001659 8B1D[966F0000]     <1>       mov    ebx, [BUFFER_TAIL]  ; GET THE END POINTER TO THE BUFFER
1290 0000165F 89DE               <1>       mov    esi, ebx            ; SAVE THE VALUE
1291 00001661 E857FAFFFF         <1>       call   _K4                 ; ADVANCE THE TAIL
1292 00001666 3B1D[926F0000]     <1>       cmp    ebx, [BUFFER_HEAD]  ; HAS THE BUFFER WRAPPED AROUND
1293 0000166C 740E               <1>       je     short K62           ; BUFFER_FULL_BEEP
1294 0000166E 668906             <1>       mov    [esi], ax           ; STORE THE VALUE
1295 00001671 891D[966F0000]     <1>       mov    [BUFFER_TAIL], ebx  ; MOVE THE POINTER UP
1296 00001677 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
1300                             <1>       ;MOV   AL, ENA_KBD         ; INSURE KEYBOARD IS ENABLED
1301                             <1>       ;CALL  SHIP_IT                  ; EXECUTE ENABLE
1302                             <1>       ;MOV   AX, 9102H           ; MOVE IN POST CODE & TYPE
1303                             <1>       ;INT   15H                 ; PERFORM OTHER FUNCTION
1304                             <1>       ;;and  byte [KB_FLAG_3],~(LC_E0+LC_E1) ; RESET LAST CHAR H.C. FLAG
1305                             <1>       ;JMP   K27A                ; INTERRUPT_RETURN
1306                             <1>       ;;jmp   K27
1307                             <1>       ;
```

```
1308                            <1>     ;----- BUFFER IS FULL SOUND THE BEEPER
1309                            <1> K62:
1310 0000167C B020              <1>     mov   al, EOI                   ; ENABLE INTERRUPT CONTROLLER CHIP
1311 0000167E E620              <1>     out   INTA00, al
1312 00001680 66B9A602          <1>     mov   cx, 678                   ; DIVISOR FOR 1760 HZ
1313 00001684 B304              <1>     mov   bl, 4                ; SHORT BEEP COUNT (1/16 + 1/64 DELAY)
1314 00001686 E8650D0000        <1>     call  beep                 ; GO TO COMMON BEEP HANDLER
1315 0000168B E901FDFFFF        <1>     jmp    K27                 ; EXIT
1316                            <1>
1317                            <1> SHIP_IT:
1318                            <1>     ;--------------------------------------------------------------------------------
1319                            <1>     ; SHIP_IT
1320                            <1>     ;     THIS ROUTINES HANDLES TRANSMISSION OF COMMAND AND DATA BYTES
1321                            <1>     ;     TO THE KEYBOARD CONTROLLER.
1322                            <1>     ;--------------------------------------------------------------------------------
1323                            <1>     ;
1324 00001690 6650              <1>     push  ax                   ; SAVE DATA TO SEND
1325                            <1>
1326                            <1>     ;----- WAIT FOR COMMAND TO ACCEPTED
1327 00001692 FA                <1>     cli                        ; DISABLE INTERRUPTS TILL DATA SENT
1328                            <1>     ; xor  ecx, ecx            ; CLEAR TIMEOUT COUNTER
1329 00001693 B900000100        <1>     mov   ecx, 10000h
1330                            <1> S10:
1331 00001698 E464              <1>     in    al, STATUS_PORT           ; READ KEYBOARD CONTROLLER STATUS
1332 0000169A A802              <1>     test  al, INPT_BUF_FULL    ; CHECK FOR ITS INPUT BUFFER BUSY
1333 0000169C E0FA              <1>     loopnz S10                 ; WAIT FOR COMMAND TO BE ACCEPTED
1334                            <1>
1335 0000169E 6658              <1>     pop   ax                   ; GET DATA TO SEND
1336 000016A0 E664              <1>     out   STATUS_PORT, al           ; SEND TO KEYBOARD CONTROLLER
1337 000016A2 FB                <1>     sti                        ; ENABLE INTERRUPTS AGAIN
1338 000016A3 C3                <1>     retn                       ; RETURN TO CALLER
1339                            <1>
1340                            <1> SND_DATA:
1341                            <1>     ; --------------------------------------------------------------------------------
1342                            <1>     ; SND_DATA
1343                            <1>     ;     THIS ROUTINES HANDLES TRANSMISSION OF COMMAND AND DATA BYTES
1344                            <1>     ;     TO THE KEYBOARD AND RECEIPT OF ACKNOWLEDGEMENTS. IT ALSO
1345                            <1>     ;     HANDLES ANY RETRIES IF REQUIRED
1346                            <1>     ; --------------------------------------------------------------------------------
1347                            <1>     ;
1348 000016A4 6650              <1>     push  ax                   ; SAVE REGISTERS
1349 000016A6 6653              <1>     push  bx
1350 000016A8 51                <1>     push  ecx
1351 000016A9 88C7              <1>     mov   bh, al               ; SAVE TRANSMITTED BYTE FOR RETRIES
1352 000016AB B303              <1>     mov   bl, 3                ; LOAD RETRY COUNT
1353                            <1> SD0:
1354 000016AD FA                <1>     cli                        ; DISABLE INTERRUPTS
1355 000016AE 8025[876F0000]CF  <1>     and   byte [KB_FLAG_2], ~(KB_FE+KB_FA) ; CLEAR ACK AND RESEND FLAGS
1356                            <1>     ;
1357                            <1>     ;----- WAIT FOR COMMAND TO BE ACCEPTED
1358 000016B5 B900000100        <1>     mov   ecx, 10000h          ; MAXIMUM WAIT COUNT
1359                            <1> SD5:
1360 000016BA E464              <1>     in    al, STATUS_PORT           ; READ KEYBOARD PROCESSOR STATUS PORT
1361 000016BC A802              <1>     test  al, INPT_BUF_FULL    ; CHECK FOR ANY PENDING COMMAND
1362 000016BE E0FA              <1>     loopnz SD5                 ; WAIT FOR COMMAND TO BE ACCEPTED
1363                            <1>     ;
1364 000016C0 88F8              <1>     mov   al, bh               ; REESTABLISH BYTE TO TRANSMIT
1365 000016C2 E660              <1>     out   PORT_A, al           ; SEND BYTE
1366 000016C4 FB                <1>     sti                        ; ENABLE INTERRUPTS
1367                            <1>     ;mov   cx, 01A00h          ; LOAD COUNT FOR 10 ms+
1368 000016C5 B9FFFF0000        <1>     mov   ecx, 0FFFFh
1369                            <1> SD1:
1370 000016CA F605[876F0000]30  <1>     test  byte [KB_FLAG_2], KB_FE+KB_FA ; SEE IF EITHER BIT SET
1371 000016D1 750F              <1>     jnz   short SD3            ; IF SET, SOMETHING RECEIVED GO PROCESS
1372 000016D3 E2F5              <1>     loop  SD1                  ; OTHERWISE WAIT
1373                            <1> SD2:
1374 000016D5 FECB              <1>     dec   bl                   ; DECREMENT RETRY COUNT
1375 000016D7 75D4              <1>     jnz   short SD0            ; RETRY TRANSMISSION
1376 000016D9 800D[876F0000]80  <1>     or    byte [KB_FLAG_2], KB_ERR ; TURN ON TRANSMIT ERROR FLAG
1377 000016E0 EB09              <1>     jmp   short SD4            ; RETRIES EXHAUSTED FORGET TRANSMISSION
1378                            <1> SD3:
1379 000016E2 F605[876F0000]10  <1>     test  byte [KB_FLAG_2], KB_FA ; SEE IF THIS IS AN ACKNOWLEDGE
1380 000016E9 74EA              <1>     jz    short SD2            ; IF NOT, GO RESEND
1381                            <1> SD4:
1382 000016EB 59                <1>     pop   ecx                  ; RESTORE REGISTERS
1383 000016EC 665B              <1>     pop   bx
1384 000016EE 6658              <1>     pop   ax
1385 000016F0 C3                <1>     retn                       ; RETURN, GOOD TRANSMISSION
1386                            <1>
1387                            <1> SND_LED:
1388                            <1>     ; --------------------------------------------------------------------------------
1389                            <1>     ; SND_LED
1390                            <1>     ;     THIS ROUTINES TURNS ON THE MODE INDICATORS.
1391                            <1>     ;
1392                            <1>     ;--------------------------------------------------------------------------------
1393                            <1>     ;
1394 000016F1 FA                <1>     cli                        ; TURN OFF INTERRUPTS
1395 000016F2 F605[876F0000]40  <1>     test  byte [KB_FLAG_2], KB_PR_LED ; CHECK FOR MODE INDICATOR UPDATE
1396 000016F9 755F              <1>     jnz   short SL1            ; DON'T UPDATE AGAIN IF UPDATE UNDERWAY
1397                            <1>     ;
1398 000016FB 800D[876F0000]40  <1>     or    byte [KB_FLAG_2], KB_PR_LED ; TURN ON UPDATE IN PROCESS
1399 00001702 B020              <1>     mov   al, EOI                   ; END OF INTERRUPT COMMAND
1400 00001704 E620              <1>     out   20h, al ;out INTA00, al   ; SEND COMMAND TO INTERRUPT CONTROL PORT
1401 00001706 EB11              <1>     jmp   short SL0            ; GO SEND MODE INDICATOR COMMAND
1402                            <1> SND_LED1:
1403 00001708 FA                <1>     cli                        ; TURN OFF INTERRUPTS
1404 00001709 F605[876F0000]40  <1>     test  byte [KB_FLAG_2], KB_PR_LED ; CHECK FOR MODE INDICATOR UPDATE
1405 00001710 7548              <1>     jnz   short SL1            ; DON'T UPDATE AGAIN IF UPDATE UNDERWAY
1406                            <1>     ;
1407 00001712 800D[876F0000]40  <1>     or    byte [KB_FLAG_2], KB_PR_LED ; TURN ON UPDATE IN PROCESS
1408                            <1> SL0:
1409 00001719 B0ED              <1>     mov   al, LED_CMD          ; LED CMD BYTE
1410 0000171B E884FFFFFF        <1>     call  SND_DATA             ; SEND DATA TO KEYBOARD
1411 00001720 FA                <1>     cli
1412 00001721 E836000000        <1>     call  MAKE_LED             ; GO FORM INDICATOR DATA BYTE
```

```
1413 00001726 8025[876F0000]F8    <1>       and   byte [KB_FLAG_2], 0F8h    ; ~KB_LEDS ; CLEAR MODE INDICATOR BITS
1414 0000172D 0805[876F0000]      <1>       or    [KB_FLAG_2], al     ; SAVE PRESENT INDICATORS FOR NEXT TIME
1415 00001733 F605[876F0000]80    <1>       test  byte [KB_FLAG_2], KB_ERR ; TRANSMIT ERROR DETECTED
1416 0000173A 750F                <1>       jnz   short SL2           ; IF SO, BYPASS SECOND BYTE TRANSMISSION
1417                              <1>       ;
1418 0000173C E863FFFFFF          <1>       call  SND_DATA            ; SEND DATA TO KEYBOARD
1419 00001741 FA                  <1>       cli                       ; TURN OFF INTERRUPTS
1420 00001742 F605[876F0000]80    <1>       test  byte [KB_FLAG_2], KB_ERR ; TRANSMIT ERROR DETECTED
1421 00001749 7408                <1>       jz    short SL3           ; IF NOT, DON'T SEND AN ENABLE COMMAND
1422                              <1> SL2:
1423 0000174B B0F4                <1>       mov   al, KB_ENABLE       ; GET KEYBOARD CSA ENABLE COMMAND
1424 0000174D E852FFFFFF          <1>       call  SND_DATA            ; SEND DATA TO KEYBOARD
1425 00001752 FA                  <1>       cli                       ; TURN OFF INTERRUPTS
1426                              <1> SL3:
1427 00001753 8025[876F0000]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 0000175A FB                  <1>       sti                       ; ENABLE INTERRUPTS
1430 0000175B 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 0000175C A0[856F0000]        <1>       mov   al, [KB_FLAG]       ; GET CAPS & NUM LOCK INDICATORS
1441 00001761 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 00001763 C0C004              <1>       rol   al, 4 ; 20/02/2015
1445 00001766 2407                <1>       and   al, 07h                   ; MAKE SURE ONLY MODE BITS ON
1446                              <1>       ;pop   cx
1447 00001768 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 ///
2653
2654                                  %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: 24/01/2021
   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 '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 00001769 9C                  <1>       pushfd
  35 0000176A 0E                  <1>       push  cs
  36 0000176B E851000000          <1>       call  VIDEO_IO_1
  37 00001770 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                           :
  56                              <1> ;                     -CURSOR IS NOT DISPLAYED IN GRAPHICS MODE       :
  57                              <1> ;    (AH)= 01H     SET CURSOR TYPE                                           :
  58                              <1> ;           (CH) = BITS 4-0 = START LINE FOR CURSOR                   :
  59                              <1> ;                  ** HARDWARE WILL ALWAYS CAUSE BLINK                :
  60                              <1> ;                  ** SETTING BIT 5 OR 6 WILL CAUSE ERRATIC BLINKING  :
  61                              <1> ;                     OR NO CURSOR AT ALL                             :
  62                              <1> ;           (CL) = BITS 4-0 = END LINE FOR CURSOR                     :
  63                              <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>
00001771 [221B0000]     <1> M1:   dd    SET_MODE     ; TABLE OF ROUTINES WITHIN VIDEO I/O
00001775 [EF1E0000]     <1>       dd    SET_CTYPE
00001779 [231F0000]     <1>       dd    SET_CPOS
0000177D [4B1F0000]     <1>       dd    READ_CURSOR
                        <1>       ;dd   VIDEO_RETURN ; READ_LPEN
00001781 [201B0000]     <1>       dd    set_mode_ncm ; Set mode without clearing video memory
00001785 [911F0000]     <1>       dd    ACT_DISP_PAGE
00001789 [23200000]     <1>       dd    SCROLL_UP
0000178D [4D210000]     <1>       dd    SCROLL_DOWN
00001791 [CD210000]     <1>       dd    READ_AC_CURRENT
00001795 [2A220000]     <1>       dd    WRITE_AC_CURRENT
00001799 [50220000]     <1>       dd    WRITE_C_CURRENT
0000179D [8B2B0000]     <1>       dd    SET_COLOR
000017A1 [F62B0000]     <1>       dd    WRITE_DOT
000017A5 [C12B0000]     <1>       dd    READ_DOT
000017A9 [E0220000]     <1>       dd    WRITE_TTY
000017AD [081B0000]     <1>       dd    VIDEO_STATE
000017B1 [8A360000]     <1>       dd    vga_pal_funcs; 10/08/2016 (TRDOS 386)
000017B5 [03310000]     <1>       dd    font_setup   ; 10/07/2016 (TRDOS 386)
000017B9 [571B0000]     <1>       dd    VIDEO_RETURN ; RESERVED
000017BD [51240000]     <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
000017C1 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
000017C2 80FC14         <1>       cmp    ah, M1L/4
000017C5 7205           <1>       jb     short VGA_func
                        <1>
000017C7 80FC4F         <1>       cmp    ah, 4Fh
000017CA 7532           <1>       jne    short M4 ; invalid !
                        <1>
                        <1> VGA_func: ; 26/11/2020
000017CC 06             <1>       push   es ; *
000017CD 1E             <1>       push   ds ; **                   ; SAVE WORK AND PARAMETER REGISTERS
                        <1>
                        <1>       ; 26/11/2020
000017CE 50             <1>       push   eax ; -
                        <1>
000017CF 66B81000       <1>       mov    ax, KDATA          ; POINT DS: TO DATA SEGMENT
000017D3 8ED8           <1>       mov    ds, ax
000017D5 8EC0           <1>       mov    es, ax
                        <1>
                        <1>       ; 26/11/2020
000017D7 58             <1>       pop    eax ; +
                        <1>       ;
000017D8 FB             <1>       sti
000017D9 80FC4F         <1>       cmp    ah, 4Fh
000017DC 747A           <1>       je     short VBE_func
                        <1>
                        <1>       ; 04/12/2020
000017DE A3[188E0100]   <1>       mov    [video_eax], eax
                        <1>
                        <1>       ; 21/12/2020
000017E3 803D[BA6F0000]FF <1>     cmp    byte [CRT_MODE], 0FFh     ; Current mode is a VESA VBE mode ?
000017EA 7213           <1>       jb     short VGA_func_std
                        <1>
000017EC 08E4           <1>       or     ah, ah             ; set mode ?
000017EE 750F           <1>       jnz    short VGA_func_std  ; no
                        <1>
000017F0 803D[54090000]03 <1>     cmp    byte [vbe3], 3            ; (real) VESA VBE 3 video bios ?
000017F7 7506           <1>       jne    short VGA_func_std  ; no
                        <1>
```

```
000017F9 E989010000          <1>       jmp    vesa_vbe3_pmi
                             <1>
                             <1>       ; 21/12/2020
                             <1> M4:                           ; COMMAND NOT VALID
000017FE CF                  <1>       iretd                   ; DO NOTHING IF NOT IN VALID RANGE
                             <1>
                             <1> VGA_func_std:
                             <1>       ; 06/12/2020
                             <1>       ; 03/12/2020
000017FF 80FC0F              <1>       cmp    ah, 0Fh
00001802 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
00001804 803D[1C120300]00    <1>       cmp    byte [pmi32], 0          ; 32 bit protected mode interface for
0000180B 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
0000180D 20E4                <1>       and    ah, ah ; is this set mode ?
0000180F 7413                <1>       jz     short vbe_pmi32_2 ; yes
                             <1>
00001811 80FC04              <1>       cmp    ah, 04h       ; set mode ('no clear memory' option)
00001814 742B                <1>       je     short VGA_funcs_0
                             <1>
                             <1>       ; 07/12/2020
00001816 803D[BA6F0000]07    <1>       cmp    byte [CRT_MODE], 7 ; current mode > 7 ?
0000181D 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:
0000181F 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
00001824 803D[BA6F0000]07    <1>       cmp    byte [CRT_MODE], 7 ; current mode > 7 ?
0000182B 77F2                <1>       ja     short vbe_pmi32_1  ; yes
                             <1>
0000182D 3C07                <1>       cmp    al, 7  ; requested mode > 7 ?
0000182F 7610                <1>       jna    short VGA_funcs_0  ; no (CGA)
                             <1>
00001831 3C13                <1>       cmp    al, 13h
00001833 76EA                <1>       jna    short vbe_pmi32_1
                             <1>
00001835 A880                <1>       test   al, 80h
00001837 7408                <1>       jz     short VGA_funcs_0  ; unknown or special
                             <1>
00001839 3C87                <1>       cmp    al, 87h       ; requested mode > 7 ?
                             <1>                            ; (with no clear mem ops)
0000183B 7604                <1>       jna    short VGA_funcs_0  ; no (CGA)
                             <1>
0000183D 3C93                <1>       cmp    al, 93h
0000183F 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:
00001841 52                  <1>       push   edx ; ***
00001842 51                  <1>       push   ecx ; ****
00001843 53                  <1>       push   ebx ; *****
00001844 56                  <1>       push   esi ; ******
00001845 57                  <1>       push   edi ; *******
00001846 55                  <1>       push   ebp ; ********
                             <1>
                             <1>       ;mov   [video_eax], eax ; 12/05/2016
00001847 BF00800B00          <1>       mov    edi, 0B8000h       ; GET offset FOR COLOR CARD
                             <1>
                             <1>       ; 23/03/2016
0000184C C0E402              <1>       shl    ah, 2  ; dword            ; TIMES 2 FOR WORD TABLE LOOKUP
0000184F 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>
```

```
00001852 FFA6[71170000]      <1>       JMP    dword [esi+M1]            ; GO TO SELECTED FUNCTION
                             <1>
                             <1> VBE_func:
                             <1>       ; 26/11/2020
                             <1>       ;sti
00001858 55                  <1>       push   ebp ; *** ; 27/11/2020
00001859 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>
0000185A 803D[54090000]02    <1>       cmp    byte [vbe3], 2
00001861 7741                <1>       ja     short VESA_VBE3_PMI_CALL ; VBE3 video bios ('PMID')
                             <1>       ;je    short VBE_func_0  ; Bochs/Qemu/VirtualBox emulator
00001863 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>
00001865 8A25[55090000]      <1>       mov    ah, [vbe2bios]
0000186B 80FCC0              <1>       cmp    ah, 0C0h
0000186E 725F                <1>       jb     short VBE_unknown
00001870 80FCC5              <1>       cmp    ah, 0C5h
00001873 775A                <1>       ja     short VBE_unknown
                             <1>
                             <1>       ; TRDOS 386 is running on BOCHS or QEMU
00001875 B44F                <1>       mov    ah, 4Fh
                             <1> VBE_func_1:
00001877 0FB6F0              <1>       movzx  esi, al ; VESA VBE function number
0000187A 66C1E602            <1>       shl    si, 2 ; dword
0000187E 6683FE14            <1>       cmp    si, N1L
00001882 734B                <1>       jnb    short VBE_unknown
                             <1>       ;sti
                             <1>
00001884 FF96[90180000]      <1>       call   dword [esi+N1] ; call VBE function
                             <1>
                             <1>       ;jmp   short VBE_bios_return
                             <1>
                             <1> VBE_bios_return:
0000188A FA                  <1>       cli
0000188B 5E                  <1>       pop    esi ; ****
0000188C 5D                  <1>       pop    ebp ; *** ; 27/11/2020
0000188D 07                  <1>       pop    es  ; **
0000188E 1F                  <1>       pop    ds  ; *
0000188F CF                  <1>       iretd
                             <1>
                             <1>       ; 26/11/2020
                             <1> N1:
00001890 [F3180000]          <1>       dd     vbe_biosfn_return_ctrl_info
00001894 [163A0000]          <1>       dd     vbe_biosfn_return_mode_info
00001898 [D13A0000]          <1>       dd     vbe_biosfn_set_mode
0000189C [A13B0000]          <1>       dd     vbe_biosfn_return_current_mode
000018A0 [C03B0000]          <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> 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>
000018A4 0FB6F0              <1>       movzx  esi, al ; VESA VBE 3 function number
000018A7 66C1E602            <1>       shl    si, 2 ; dword
000018AB 6683FE14            <1>       cmp    si, P1L
000018AF 731E                <1>       jnb    short VBE_unknown
                             <1>       ;sti
                             <1>
000018B1 57                  <1>       push   edi ; *****
                             <1>
000018B2 FF96[BB180000]      <1>       call   dword [esi+P1] ; call VBE 3 function
                             <1>
000018B8 5F                  <1>       pop    edi ; *****
                             <1>
000018B9 EBCF                <1>       jmp    short VBE_bios_return
                             <1>
                             <1> P1:
000018BB [D5180000]          <1>       dd     vbe3_pmfn_return_ctrl_info
```

```
000018BF [F9180000]           <1>       dd    vbe3_pmfn_return_mode_info
000018C3 [36190000]           <1>       dd    vbe3_pmfn_set_mode
000018C7 [31190000]           <1>       dd    vbe3_pmfn_return_current_mode
000018CB [281A0000]           <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:
000018CF 66B80001             <1>       mov   ax, 0100h  ; ah = 1 : Function call failed
                              <1>                        ; al = 0 : Function is not supported
                              <1>
000018D3 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>
000018D5 21FF                 <1>       and   edi, edi
000018D7 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:
000018D9 57             <1>       push   edi
000018DA 51             <1>       push   ecx
000018DB BE007E0900     <1>       mov    esi, VBE3INFOBLOCK
000018E0 B900020000     <1>       mov    ecx, 512
000018E5 E8DAF90000     <1>       call   transfer_to_user_buffer
000018EA 59             <1>       pop    ecx
000018EB 5F             <1>       pop    edi
000018EC 720F           <1>       jc     short vbe3_func_fail
                        <1>
000018EE 31C0           <1>       xor    eax, eax
000018F0 B04F           <1>       mov    al, 4Fh ; successful
                        <1> ;vbe_ctrl_info_retn:
000018F2 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>
000018F3 21FF           <1>       and    edi, edi
000018F5 7406           <1>       jz     short vbe3_func_fail ; invalid buffer addr !
000018F7 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>
000018F9 09FF           <1>       or     edi, edi
000018FB 7506           <1>       jnz    short _vbe3_pmfn_return_mode_info
                        <1>
                        <1> vbe3_func_fail:
000018FD B84F010000     <1>       mov    eax, 014Fh ; ah = 1 : Function call failed
                        <1>                         ; al = 4Fh : Function is supported
00001902 C3             <1>       retn
                        <1>
                        <1>       ; jump from '_vbe_biosfn_return_mode_info'
                        <1> _vbe3_pmfn_return_mode_info:
00001903 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
00001904 BF00060000     <1>       mov    edi, VBE3MODEINFOBLOCK-VBE3SAVERESTOREBLOCK
                        <1>       ; ES selector base address = VBE3SAVERESTOREBLOCK
00001909 E8FC000000     <1>       call   int10h_32bit_pmi
                        <1>
0000190E 5F             <1>       pop    edi
                        <1>
                        <1>       ;cmp   ax, 004Fh
                        <1>       ;jne   short vbe3_func_retn  ; failed !
                        <1>
0000190F 21FF           <1>       and    edi, edi
                        <1>       ;jz    short vbe3_func_success
                        <1>       ; 21/12/2020
00001911 741D           <1>       jz     short vbe3_func_retn
                        <1>
```

```
00001913 6683F84F            <1>       cmp    ax, 004Fh
00001917 7517                <1>       jne    short vbe3_func_retn  ; failed !
                             <1>
00001919 51                  <1>       push   ecx
0000191A BE007C0900          <1>       mov    esi, VBE3MODEINFOBLOCK
0000191F B900010000          <1>       mov    ecx, 256
00001924 E89BF90000          <1>       call   transfer_to_user_buffer
00001929 59                  <1>       pop    ecx
0000192A 72D1                <1>       jc     short vbe3_func_fail
                             <1>
0000192C 31C0                <1>       xor    eax, eax
0000192E B04F                <1>       mov    al, 4Fh ; successful
                             <1> vbe3_func_success:
                             <1> vbe3_func_retn:
00001930 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
00001931 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
00001936 57                  <1>       push   edi ; *****
```

```
00001937 F6C708             <1>       test   bh, 8  ; Use user specified CRTC values
0000193A 7530               <1>       jnz    short vbe3_sm_3  ; for refresh rate
                            <1> vbe3_sm_4:
0000193C 80E7C1             <1>       and    bh, 0C1h ; use bit 15, 14, 8 only (for bh)
                            <1>       ;mov   [vbe_mode_x], bh
                            <1>
0000193F 803D[BA6F0000]03   <1>       cmp    byte [CRT_MODE], 3 ; is current mode 03h ?
00001946 7509               <1>       jne    short vbe3_sm_1    ; no
                            <1>
                            <1>       ; save mode 03h video pages and cursor positions
00001948 57                 <1>       push   edi ; **!***
00001949 51                 <1>       push   ecx ; ******
                            <1>       ;push  esi
                            <1>
0000194A E8CE040000         <1>       call   save_mode3_multiscreen
                            <1>
                            <1>       ;pop   esi
0000194F 59                 <1>       pop    ecx ; ******
00001950 5F                 <1>       pop    edi ; **!***
                            <1> vbe3_sm_1:
                            <1>       ; ax = 4F02h
                            <1>       ; bx = video mode number (vbe2 type)
00001951 E8B4000000         <1>       call   int10h_32bit_pmi
                            <1>                     ; call to far call to VBE3 PMI
                            <1>
00001956 6683F84F           <1>       cmp    ax, 004Fh ; succeeded ?
0000195A 750E               <1>       jne    short vbe3_sm_2
                            <1>       ; set current mode byte/sign to extended (SVGA) mode
0000195C C605[BA6F0000]FF   <1>       mov    byte [CRT_MODE], 0FFh ; VESA VBE mode
                            <1>       ; set current VBE mode word to bx input
00001963 66891D[1E120300]   <1>       mov    [video_mode], bx
                            <1> vbe3_sm_2:
                            <1>       ; 22/12/2020
0000196A 5F                 <1>       pop    edi ; *****
0000196B C3                 <1>       retn
                            <1>
                            <1> vbe3_sm_3:
                            <1>       ; 22/12/2020
                            <1>       ; copy user's CRTCInfoBlock to the buffer
0000196C 51                 <1>       push   ecx
0000196D 89FE               <1>       mov    esi, edi
0000196F BF807D0900         <1>       mov    edi, VBE3CRTCINFOBLOCK
00001974 B940000000         <1>       mov    ecx, 64
00001979 E890F90000         <1>       call   transfer_from_user_buffer
0000197E 59                 <1>       pop    ecx
                            <1>       ; set offset (es base addr is VBE3SAVERESTOREBLOCK)
0000197F 81EF00760900       <1>       sub    edi, VBE3SAVERESTOREBLOCK
00001985 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
00001987 20E4               <1>       and    ah, ah ; 0 = set mode function
00001989 7402               <1>       jz     short vbe3_pmi_0
0000198B EB76               <1>       jmp    vbe3_pmi_9
                            <1>
                            <1> vbe3_pmi_0:
                            <1>       ; 07/12/2020
0000198D 88C4               <1>       mov    ah, al
0000198F 80E480             <1>       and    ah, 80h ; 0 or 80h
00001992 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
00001994 8825[278E0100]     <1>       mov    byte [noclearmem], ah ; 0 or 80h
                            <1>       ; check current video mode if it is 03h
0000199A 803D[BA6F0000]03   <1>       cmp    byte [CRT_MODE], 3 ; current mode
000019A1 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
000019A3 56                 <1>       push   esi
000019A4 57                 <1>       push   edi
000019A5 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
000019A6 E872040000             <1>       call   save_mode3_multiscreen
                                <1>
000019AB 59                     <1>       pop    ecx
000019AC 5F                     <1>       pop    edi
000019AD 5E                     <1>       pop    esi
                                <1> vbe3_pmi_3:
                                <1>       ; 08/12/2020
                                <1>       ; 07/12/2020
                                <1>       ; case 3 or case 4
000019AE A2[BA6F0000]           <1>       mov    [CRT_MODE], al
000019B3 3C03                   <1>       cmp    al, 3
000019B5 7407                   <1>       je     short vbe3_pmi_4
                                <1>       ; case 4:
                                <1>       ; Current mode is not 03h and also new mode is not 03h
000019B7 800D[258E0100]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>
000019BE E847000000             <1>       call   int10h_32bit_pmi
                                <1>
000019C3 803D[BA6F0000]03       <1>       cmp    byte [CRT_MODE], 3  ; new video mode
                                <1>       ;jne   vbe3_pmi_8   ; video mode <> 03h
000019CA 7532                   <1>       jne    short vbe3_pmi_8
                                <1>
                                <1>       ;push  eax ; 04/12/2020
000019CC 53                     <1>       push   ebx
000019CD 51                     <1>       push   ecx
000019CE 52                     <1>       push   edx
000019CF 57                     <1>       push   edi ; 03/12/2020
                                <1>
                                <1>       ; 12/12/2020
000019D0 56                     <1>       push   esi
000019D1 E87A040000             <1>       call   restore_mode3_multiscreen
000019D6 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>       ;
                                <1>       ;; 07/12/2020
                                <1>       ;; restore CRT_START according to ACTIVE_PAGE
                                <1>       ;mov   [CRT_START], cx ; 0
                                <1>       ;
                                <1>       ;; check active page and set it again if it is not 0
                                <1>       ;or    al, al
                                <1>       ;jz    short vbe3_pmi_7
                                <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 000019D7 B405                      <1>       mov    ah, 05h ; set current video page
1016                                    <1>       ;al = video page
1017 000019D9 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 000019DE 0FB6D8                    <1>       movzx  ebx, al
1022 000019E1 D0E3                      <1>       shl    bl, 1
1023 000019E3 81C3[AE810100]            <1>       add    ebx, CURSOR_POSN
1024 000019E9 668B13                    <1>       mov    dx, [ebx]
1025 000019EC 6621D2                    <1>       and    dx, dx
1026 000019EF 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 000019F1 88C7                      <1>       mov    bh, al
1031 000019F3 B402                      <1>       mov    ah, 02h ; set cursor position
1032 000019F5 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 000019FA 5F                        <1>       pop    edi
1051 000019FB 5A                        <1>       pop    edx
1052 000019FC 59                        <1>       pop    ecx
1053 000019FD 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 000019FE 31C0                      <1>       xor    eax, eax  ; eax = 0 -> succesful
1059                                    <1> vesa_vbe3_pmi_retn:
1060 00001A00 07                        <1>       pop    es  ; **
1061 00001A01 1F                        <1>       pop    ds  ; *
1062 00001A02 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 00001A03 E802000000                <1>       call   int10h_32bit_pmi
1070 00001A08 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 00001A0A 56                        <1>       push   esi
1089 00001A0B C1E010                    <1>       shl    eax, 16  ; move function number (ax) to hw
1090 00001A0E 8B35[24120300]            <1>       mov    esi, [pmid_addr] ; linear address of
1091                                    <1>                               ; PMInfo.Entrypoint pointer
1092                                    <1>       ;mov   ax, [esi+PMInfo.EntryPoint]
1093 00001A14 668B06                    <1>       mov    ax, [esi]
1094 00001A17 C1C010                    <1>       rol    eax, 16 ; move PM entry address to hw
1095                                    <1>                      ; and move function number to lw (ax)
1096 00001A1A 5E                        <1>       pop    esi
1097                                    <1>
1098                                    <1>       ; top of stack: ; (*)
1099                                    <1>       ; return (the caller) address of "int10h_32bit_pmi"
1100                                    <1>
1101 00001A1B E97CEDFFFF                <1>       jmp    _VBE3PMI_fcall ; will return to the caller (*)
1102                                    <1>
1103                                    <1> _vbe3_pmfn_srs_8:
1104                                    <1>       ; 17/01/2021
1105 00001A20 31DB                      <1>       xor    ebx, ebx ; points to VBE3SAVERESTOREBLOCK
1106                                    <1> _vbe3_pmfn_srs_9: ; 24/01/2021
1107 00001A22 66B8044F                  <1>       mov    ax, 4F04h
1108 00001A26 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
```

```
                        <1>       ; 14/01/2021
                        <1>       ;
                        <1>       ; VBE function 4F04h - Save/Restore Video State
                        <1>       ;
                        <1>       ; Input:
                        <1>       ;     DL = sub function
                        <1>       ;     CL = requested state
                        <1>       ;      EBX = pointer to buffer (if DL<>00h)
                        <1>       ;     AX = 4F04h
                        <1>       ; Output:
                        <1>       ;     AX = 004Fh (successful)
                        <1>       ;     AH > 0 -> error
                        <1>       ;     BX = Number of 64-byte blocks
                        <1>       ;          to hold the state buffer (if DL=00h)
                        <1>
                        <1>       ; Modified registers: eax, ebx, esi, edi
                        <1>
00001A28 21DB           <1>       and    ebx, ebx ; user's buffer address
00001A2A 750A           <1>       jnz    short _vbe3_pmfn_save_restore_state
                        <1>
00001A2C 08D2           <1>       or     dl, dl
00001A2E 740C           <1>       jz     short _vbe3_pmfn_srs_0
                        <1>
                        <1>       ; function failed
                        <1>       ;;mov  eax, 0100h
                        <1>       ;sub   eax, eax
                        <1>       ;inc   ah  ; eax = 0100h
                        <1>       ;retn
                        <1>       ; 16/01/2021
                        <1> _vbe3_pmfn_srs_fail:
00001A30 B84F010000     <1>       mov    eax, 014Fh ; ah = 1 : Function call failed
                        <1>                         ; al = 4Fh : Function is supported
                        <1> _vbe3_srs_retn:
00001A35 C3             <1>       retn
                        <1>
                        <1> _vbe3_pmfn_save_restore_state:
00001A36 20D2           <1>       and    dl, dl
00001A38 7559           <1>       jnz    short _vbe3_pmfn_srs_2
                        <1> _vbe3_pmfn_srs:
00001A3A 31DB           <1>       xor    ebx, ebx
                        <1> _vbe3_pmfn_srs_0:
                        <1>       ; 24/01/2021
00001A3C 83F90F         <1>       cmp    ecx, 0Fh
                        <1>       ;ja    short _vbe3_pmfn_srs_1
00001A3F 77EF           <1>       ja     short _vbe3_pmfn_srs_fail
                        <1>
                        <1>       ; !!! CLEAR CL BIT 2 !!!
                        <1>       ; (when bit 2 is set, function causes cpu exception)
                        <1>       ; BIOS data will not be saved and restored
                        <1>       ; (to prevent protected mode page fault error)
00001A41 80E1FD         <1>       and    cl, ~2 ; and cl, not 2
                        <1>
                        <1>       ; 24/01/2021
                        <1>       ;mov   bl, 1
00001A44 FEC3           <1>       inc    bl ; = 1
00001A46 66D3E3         <1>       shl    bx, cl
00001A49 66231D[28120300]   <1>   and    bx, [vbe3stbsflags]
00001A50 7416           <1>       jz     short _vbe3_pmfn_srs_1
                        <1>       ;mov   bx, cx
00001A52 89CB           <1>       mov    ebx, ecx ; <= 15
00001A54 D0E3           <1>       shl    bl, 1 ; 0, 2, 8 .. 30
00001A56 668B9B[713C0000]   <1>   mov    bx, [vbestatebufsize+ebx]
00001A5D 89DF           <1>       mov    edi, ebx
                        <1>       ; edi = state buffer size in bytes
00001A5F 66C1EB06       <1>       shr    bx, 6 ; / 64
00001A63 66B84F00       <1>       mov    ax, 4Fh
00001A67 C3             <1>       retn
                        <1> _vbe3_pmfn_srs_1:
                        <1>       ; ax = 4F04h
                        <1>       ;call  int10h_32bit_pmi
                        <1>       ; 24/01/2021
                        <1>       ;call  _vbe3_pmfn_srs_8
                        <1>       ; ebx = 0
00001A68 E8B5FFFFFF     <1>       call   _vbe3_pmfn_srs_9
00001A6D 6683F84F       <1>       cmp    ax, 004Fh
00001A71 75C2           <1>       jne    short _vbe3_srs_retn
                        <1>       ; 24/01/2021
                        <1>       ;cmp   ecx, 0Fh
                        <1>       ;ja    short _vbe3_srs_retn
                        <1>       ; 24/01/2021
                        <1>       ;mov   ax, 1
00001A73 B001           <1>       mov    al, 1
00001A75 66D3E0         <1>       shl    ax, cl
00001A78 660905[28120300]   <1>   or     [vbe3stbsflags], ax ; set flag for state option
                        <1>       ; 23/01/2021
00001A7F 89DF           <1>       mov    edi, ebx
00001A81 89C8           <1>       mov    eax, ecx
00001A83 D0E0           <1>       shl    al, 1
00001A85 66C1E706       <1>       shl    di, 6 ; * 64
00001A89 6689B8[713C0000]   <1>   mov    [vbestatebufsize+eax], di
                        <1>                           ; save buf size for option
                        <1>       ;xchg  edi, ebx
                        <1>       ; edi = state buffer size in bytes
00001A90 B04F           <1>       mov    al, 4Fh
00001A92 C3             <1>       retn
                        <1>
                        <1> _vbe3_pmfn_srs_2:
                        <1>       ; 24/01/2021
                        <1>       ; !!! CLEAR CL BIT 2 !!!
                        <1>       ; (when bit 2 is set, function causes cpu exception)
                        <1>       ; BIOS data will not be saved and restored
                        <1>       ; (to prevent protected mode page fault error)
                        <1>
00001A93 F6C1FD         <1>       test   cl, ~2 ; test cl, not 2
00001A96 7498           <1>       jz     short _vbe3_pmfn_srs_fail
```

```
1219                                    <1>
1220 00001A98 80FA02                    <1>       cmp    dl, 2
1221 00001A9B 7748                      <1>       ja     short _vbe3_pmfn_srs_5
1222                                    <1>
1223                                    <1>       ;and   cl, ~2 ; and cl, not 2
1224                                    <1>
1225 00001A9D 53                        <1>       push   ebx ; * ; buffer address
1226                                    <1>       ; save or restore state
1227                                    <1>       ; (get required buffer size at first)
1228 00001A9E 52                        <1>       push   edx ; **
1229 00001A9F 28D2                      <1>       sub    dl, dl ; 0
1230 00001AA1 E894FFFFFF                <1>       call   _vbe3_pmfn_srs
1231 00001AA6 5A                        <1>       pop    edx ; **
1232                                    <1>       ; 24/01/2021
1233 00001AA7 5B                        <1>       pop    ebx ; *
1234 00001AA8 08E4                      <1>       or     ah, ah
1235 00001AAA 7538                      <1>       jnz    short _vbe3_pmfn_srs_4 ; error
1236                                    <1>
1237                                    <1>       ; edi = buffer size in bytes
1238 00001AAC 81FF00080000              <1>       cmp    edi, 2048
1239 00001AB2 772B                      <1>       ja     short _vbe3_pmfn_srs_3
1240                                    <1>
1241 00001AB4 80FA01                    <1>       cmp    dl, 1
1242 00001AB7 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 00001AB9 E842000000                <1>       call   _vbe3_pmfn_srs_7
1251                                    <1>
1252 00001ABE 6683F84F                  <1>       cmp    ax, 004Fh
1253 00001AC2 7520                      <1>       jne    short _vbe3_pmfn_srs_4
1254                                    <1>
1255 00001AC4 09DB                      <1>       or     ebx, ebx  ; kernel ('sysvideo') ?
1256 00001AC6 741C                      <1>       jz     short _vbe3_pmfn_srs_4 ; yes
1257                                    <1>
1258                                    <1>       ; the caller is user
1259 00001AC8 51                        <1>       push   ecx ; *
1260 00001AC9 89F9                      <1>       mov    ecx, edi ; state buffer size
1261 00001ACB BE00760900                <1>       mov    esi, VBE3SAVERESTOREBLOCK ; source
1262                                    <1>                                 ; (vbe3 pmi buff)
1263 00001AD0 89DF                      <1>       mov    edi, ebx     ; destination (user buff)
1264 00001AD2 E8EDF70000                <1>       call   transfer_to_user_buffer
1265 00001AD7 59                        <1>       pop    ecx ; *
1266 00001AD8 7205                      <1>       jc     short _vbe3_pmfn_srs_3
1267                                    <1>
1268 00001ADA 29C0                      <1>       sub    eax, eax
1269 00001ADC B04F                      <1>       mov    al, 4Fh
1270 00001ADE C3                        <1>       retn
1271                                    <1>
1272                                    <1>       ; 24/01/2021
1273                                    <1> _vbe3_pmfn_srs_3:
1274 00001ADF B84F010000                <1>       mov    eax, 014Fh
1275                                    <1> _vbe3_pmfn_srs_4:
1276 00001AE4 C3                        <1>       retn
1277                                    <1> _vbe3_pmfn_srs_5:
1278 00001AE5 31C0                      <1>       xor    eax, eax
1279 00001AE7 FEC4                      <1>       inc    ah
1280                                    <1>       ; eax = 0100h, function is not supported
1281 00001AE9 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 00001AEA 09DB                      <1>       or     ebx, ebx ; 0 ?
1289 00001AEC 7412                      <1>       jz     short _vbe3_pmfn_srs_7 ; 'sysvideo' call
1290                                    <1>       ; 24/01/2021
1291                                    <1>       ;jz    _vbe3_pmfn_srs_8
1292 00001AEE 89DE                      <1>       mov    esi, ebx
1293                                    <1>       ; esi = user's video state buffer
1294 00001AF0 51                        <1>       push   ecx ; *
1295 00001AF1 89F9                      <1>       mov    ecx, edi ; state buffer size
1296 00001AF3 BF00760900                <1>       mov    edi, VBE3SAVERESTOREBLOCK ; destination
1297                                    <1>                                 ; (vbe3 pmi buff)
1298                                    <1>       ;mov   esi, ebx     ; source (user buff)
1299 00001AF8 E811F80000                <1>       call   transfer_from_user_buffer
1300 00001AFD 59                        <1>       pop    ecx ; *
1301 00001AFE 72DF                      <1>       jc     short _vbe3_pmfn_srs_3
1302                                    <1> _vbe3_pmfn_srs_7:
1303 00001B00 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 00001B01 E81AFFFFFF                <1>       call   _vbe3_pmfn_srs_8
1310 00001B06 5B                        <1>       pop    ebx ; *
1311 00001B07 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 00001B08 8A25[BC6F0000]            <1>         mov    ah, [CRT_COLS]      ; GET NUMBER OF COLUMNS
1327 00001B0E A0[BA6F0000]              <1>         mov    al, [CRT_MODE]      ; CURRENT MODE
1328                                    <1>         ;movzx esi, al
1329                                    <1>         ;mov   ah, [esi+M6]
1330                                    <1>         ; BH = active page
1331 00001B13 8A3D[BE810100]            <1>         mov    bh, [ACTIVE_PAGE] ; GET CURRENT ACTIVE PAGE
1332 00001B19 FA                        <1>         cli    ; 02/01/2017
1333 00001B1A 5D                        <1>         pop    ebp          ; RECOVER REGISTERS
1334 00001B1B 5F                        <1>         pop    edi
1335 00001B1C 5E                        <1>         pop    esi
1336 00001B1D 59                        <1>         pop    ecx          ; DISCARD SAVED BX
1337 00001B1E 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 00001B20 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 00001B22 803D[BA6F0000]FF          <1>         cmp    byte [CRT_MODE], 0FFh ; VESA VBE graphics mode
1378 00001B29 7220                      <1>         jb     short _set_mode_      ; signature
1379                                    <1>                                      ; VBE mode number is in
1380                                    <1>                                      ; [video_mode] bit 0to8
1381 00001B2B 88C3                      <1>         mov    bl, al ; save video mode
1382 00001B2D E893240000                <1>         call   dispi_get_enable
1383 00001B32 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 00001B33 31C0                      <1>         xor    eax, eax ; 0
1387 00001B35 E85F240000                <1>         call   dispi_set_enable
1388 00001B3A 88D8                      <1>         mov    al, bl ; restore video mode
1389 00001B3C E827000000                <1>         call   _set_mode
1390 00001B41 58                        <1>         pop    eax ; restore current VBE dispi status
1391 00001B42 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 00001B44 E850240000                <1>         call   dispi_set_enable
1399 00001B49 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 00001B4B E818000000                <1>         call   _set_mode ; 24/06/2016 (set_txt_mode)
1417                                    <1>         ; 26/11/2020
1418 00001B50 7305                      <1>         jnc    short VIDEO_RETURN
1419                                    <1>
1420                                    <1>         ; 26/11/2020
1421                                    <1> _video_func_err:
1422 00001B52 31C0                      <1>         xor    eax, eax ; function call failed
1423 00001B54 48                        <1>         dec    eax  ; 0FFFFFFFFh ; - 1
1424 00001B55 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 00001B57 A1[188E0100]      <1>       mov   eax, [video_eax] ; 12/05/2016
1433                            <1> _video_return:
1434 00001B5C FA                <1>       cli ; 02/01/2017
1435 00001B5D 5D                <1>       pop   ebp ; ******** ; 26/11/2020
1436 00001B5E 5F                <1>       pop   edi ; *******
1437 00001B5F 5E                <1>       pop   esi ; ******
1438 00001B60 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 00001B61 59                <1>       pop   ecx ; **** ; 26/11/2020
1445 00001B62 5A                <1>       pop   edx ; ***
1446 00001B63 1F                <1>       pop   ds  ; **
1447 00001B64 07                <1>       pop   es  ; *      ; RECOVER SEGMENTS
1448 00001B65 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 00001B66 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 00001B68 88C4              <1>       mov   ah, al
1493 00001B6A 80E480            <1>       and   ah, 80h
1494                            <1>       ;mov  [noclearmem], al
1495 00001B6D 8825[278E0100]    <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 00001B73 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 00001B75 3C03                      <1>     cmp   al, 03h          ; mode 3, 80x25 text, 16 colors
1540 00001B77 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 00001B79 08E4                      <1>     or    ah, ah ; 80h or 0 ('noclearmem' option)
1546 00001B7B 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 00001B7D 3805[BA6F0000]            <1>     cmp   [CRT_MODE], al      ; 03h
1554 00001B83 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 00001B85 C605[258E0100]80            <1>     mov   byte [p_crt_mode], 80h ; clear page sign
1567 00001B8C 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 00001B8E 803D[BA6F0000]03          <1>     cmp   byte [CRT_MODE], 3 ; is current mode 03h?
1573 00001B95 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 00001B97 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 00001B9C 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 00001B9E 800D[258E0100]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 00001BA5 A2[BA6F0000]               <1>     mov   [CRT_MODE], al  ; save mode in global variable
1636                                    <1> _sm_3:
1637                                    <1>     ; 30/07/2016
1638                                    <1>     ; 26/07/2016
```

```
                        <1>       ; 25/07/2016
                        <1>       ; set_mode_vga:
                        <1>       ; 18/07/2016
                        <1>       ; 14/07/2016
                        <1>       ; 09/07/2016
                        <1>       ; 04/07/2016
                        <1>       ; 03/07/2016 (TRDOS 386 = TRDOS v2.0)
                        <1>       ; /// video mode 13h ///
                        <1>       ; derived from 'Plex86/Bochs VGABios' source code
                        <1>       ; vgabios-0.7a (2011)
                        <1>       ; by the LGPL VGABios developers Team (2001-2008)
                        <1>       ; 'vgabios.c', 'vgatables.h'
                        <1>       ;
                        <1>       ; Oracle VirtualBox 5.0.24 VGABios Source Code
                        <1>       ; ('vgabios.c', 'vgatables.h', 'vgafonts.h', 'vgarom.asm')
                        <1>       ;
00001BAA 88C4           <1>       mov    ah, al
00001BAC B910000000     <1>       mov    ecx, vga_mode_count
00001BB1 BE[D66F0000]   <1>       mov    esi, vga_modes
00001BB6 31DB           <1>       xor    ebx, ebx
                        <1> _sm_4:
00001BB8 AC             <1>       lodsb
00001BB9 38C4           <1>       cmp    ah, al
00001BBB 7406           <1>       je     short _sm_5
00001BBD FEC3           <1>       inc    bl
00001BBF E2F7           <1>       loop   _sm_4
                        <1>
                        <1>       ; UNIMPLEMENTED VIDEO MODE !
                        <1>       ;xor   eax, eax
                        <1>         ;mov [video_eax], eax ; 0
                        <1>
                        <1>       ; 26/11/2020
00001BC1 F9             <1>       stc    ; unimplemented video mode ! (cf=1)
                        <1>
00001BC2 C3             <1>       retn
                        <1>
                        <1> ;-----      eBX POINTS TO CORRECT ROW OF INITIALIZATION TABLE
                        <1>
                        <1> _sm_5:      ; 25/07/2016
                        <1>       ;mov   esi, ebx
                        <1>       ;add   esi, vga_memmodel
                        <1>       ;mov   al, [esi]
                        <1>       ; 19/11/2020
00001BC3 8A83[26700000] <1>       mov    al, [ebx+vga_memmodel]
00001BC9 A2[3E8E0100]   <1>       mov    [VGA_MTYPE], al
                        <1>
00001BCE 89DF           <1>       mov    edi, ebx
00001BD0 81C7[36700000] <1>       add    edi, vga_dac_s
00001BD6 C0E302         <1>       shl    bl, 2 ; byte -> dword
00001BD9 81C3[E66F0000] <1>       add    ebx, vga_mode_tbl_ptr
                        <1>
                        <1>       ;mov   dword [VGA_BASE], 0B8000h
                        <1>       ;cmp   ah, 0Dh ; [CRT_MODE]
                        <1>       ;jb    short M9
                        <1>       ;mov   dword [VGA_BASE], 0A0000h
                        <1> ;M9:
00001BDF 8B33           <1>       mov    esi, [ebx]
00001BE1 89F3           <1>       mov    ebx, esi
00001BE3 83C614         <1>       add    esi, vga_p_cm_pos ; ebx + 20
00001BE6 668B06         <1>       mov    ax, [esi]       ; get the cursor mode from the table
00001BE9 66A3[D36F0000] <1>       mov    [CURSOR_MODE], ax ; save cursor mode (initial value)
                        <1>       ; al = 6, ah = 7
                        <1>       ; al = 0Dh, ah = 0Eh ; 25/07/2016
00001BEF E8A8020000     <1>       call   cursor_shape_fix
                        <1>       ; al = 14, ah = 15 (If [CHAR_HEIGHT] = 16)
00001BF4 668906         <1>       mov    [esi], ax
                        <1>
00001BF7 56             <1>       push   esi ; *
                        <1>
00001BF8 8A25[C16F0000] <1>       mov    ah, [VGA_MODESET_CTL]
00001BFE 80E408         <1>       and    ah, 8 ; default palette loading ?
00001C01 7524           <1>       jnz    short _sm_6
00001C03 66BAC603       <1>       mov    dx, 3C6h ; VGAREG_PEL_MASK (DAC mask register)
00001C07 B0FF           <1>       mov    al, 0FFh ; PEL mask
00001C09 EE             <1>       out    dx, al
00001C0A 8A27           <1>       mov    ah, [edi] ; DAC model (selection number)
00001C0C E874100000     <1>       call   load_dac_palette
                        <1>       ; ecx = 0
00001C11 F605[C16F0000]02 <1>     test   byte [VGA_MODESET_CTL], 2 ; gray scale summing
00001C18 740D           <1>       jz     short _sm_6
00001C1A 53             <1>       push   ebx
00001C1B 29DB           <1>       sub    ebx, ebx ; sub bl, bl
00001C1D 66B90001       <1>         mov     cx, 256
00001C21 E8B2100000     <1>       call   gray_scale_summing
00001C26 5B             <1>       pop    ebx
                        <1> _sm_6:
                        <1>       ; Reset Attribute Ctl flip-flop
00001C27 66BADA03       <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
00001C2B EC             <1>       in     al, dx
                        <1>       ; Set Attribute Ctl
00001C2C 89DE           <1>       mov    esi, ebx ; addr of params tbl for selected mode
00001C2E 83C623         <1>       add    esi, 35  ; actl regs
00001C31 30E4           <1>       xor    ah, ah ; 0
00001C33 66BAC003       <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
                        <1> _sm_7:
00001C37 88E0           <1>       mov    al, ah
00001C39 EE             <1>       out    dx, al ; index
00001C3A AC             <1>       lodsb
                        <1>       ; DX = 3C0h = VGAREG_ACTL_WRITE_DATA
00001C3B EE             <1>       out    dx, al ; value
00001C3C FEC4           <1>       inc    ah
00001C3E 80FC14         <1>       cmp    ah, 20 ; number of actl registers
00001C41 72F4           <1>       jb     short _sm_7
                        <1>       ;
00001C43 88E0           <1>       mov    al, ah ; 20
```

```
1744 00001C45 EE                <1>       out    dx, al ; index
1745 00001C46 28C0              <1>       sub    al, al ; 0
1746 00001C48 EE                <1>       out    dx, al ; value
1747                            <1>       ;
1748                            <1>       ; Set Sequencer Ctl
1749 00001C49 89DE              <1>       mov    esi, ebx ; addr of params tbl for selected mode
1750 00001C4B 83C605            <1>       add    esi, 5 ; sequ regs
1751                            <1>       ;
1752 00001C4E 66BAC403          <1>       mov    dx, 3C4h  ; VGAREG_SEQU_ADDRESS
1753 00001C52 EE                <1>       out    dx, al ; 0
1754 00001C53 6642              <1>       inc    dx ; 3C5h ; VGAREG_SEQU_DATA
1755 00001C55 B003              <1>       mov    al, 3
1756 00001C57 EE                <1>       out    dx, al
1757 00001C58 B401              <1>       mov    ah, 1
1758                            <1> _sm_8:
1759 00001C5A 88E0              <1>       mov    al, ah
1760                            <1>       ;mov   dx, 3C4h ; VGAREG_SEQU_ADDRESS
1761 00001C5C 664A              <1>       dec    dx
1762 00001C5E EE                <1>       out    dx, al ; index
1763 00001C5F AC                <1>       lodsb
1764 00001C60 6642              <1>       inc    dx ; 3C5h ; VGAREG_SEQU_DATA
1765 00001C62 EE                <1>       out    dx, al
1766 00001C63 80FC04            <1>       cmp    ah, 4 ; number of sequ regs
1767 00001C66 7304              <1>       jnb    short _sm_9
1768 00001C68 FEC4              <1>       inc    ah
1769 00001C6A EBEE              <1>       jmp    short _sm_8
1770                            <1> _sm_9:
1771                            <1>       ; Set Grafx Ctl
1772 00001C6C 89DE              <1>       mov    esi, ebx ; addr of params tbl for selected mode
1773 00001C6E 83C637            <1>       add    esi, 55 ; grdc regs
1774 00001C71 30E4              <1>       xor    ah, ah ; 0
1775                            <1> _sm_10:
1776 00001C73 88E0              <1>       mov    al, ah
1777 00001C75 66BACE03          <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
1778 00001C79 EE                <1>       out    dx, al
1779 00001C7A AC                <1>       lodsb
1780 00001C7B 6642              <1>       inc    dx ; 3CFh ; VGAREG_GRDC_DATA
1781 00001C7D EE                <1>       out    dx, al
1782 00001C7E FEC4              <1>       inc    ah
1783 00001C80 80FC09            <1>       cmp    ah, 9 ; number of grdc regs
1784 00001C83 72EE              <1>       jb     short _sm_10
1785                            <1>       ;
1786                            <1>       ; Disable CRTC write protection
1787 00001C85 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 00001C89 66B81100          <1>       mov    ax, 11h
1794 00001C8D 66EF              <1>       out    dx, ax
1795 00001C8F 89DE              <1>       mov    esi, ebx ; addr of params tbl for selected mode
1796 00001C91 83C60A            <1>       add    esi, 10 ; crtc regs
1797                            <1>       ; ah = 0
1798                            <1> _sm_11:
1799 00001C94 88E0              <1>       mov    al, ah
1800                            <1>       ; dx = 3D4h = VGAREG_VGA_CRTC_ADDRESS
1801 00001C96 EE                <1>       out    dx, al ; index
1802 00001C97 AC                <1>       lodsb
1803 00001C98 6642              <1>       inc    dx  ; VGAREG_VGA_CRTC_ADDRESS + 1
1804 00001C9A EE                <1>       out    dx, al ; value
1805 00001C9B 80FC18            <1>       cmp    ah, 24 ; number of crtc registers - 1
1806 00001C9E 7306              <1>       jnb    short _sm_12
1807 00001CA0 FEC4              <1>       inc    ah
1808 00001CA2 664A              <1>       dec    dx ; 3D4h
1809 00001CA4 EBEE              <1>       jmp    short _sm_11
1810                            <1> _sm_12:
1811                            <1>       ; Set the misc register
1812 00001CA6 66BACC03          <1>       mov    dx, 3CCh ; VGAREG_READ_MISC_OUTPUT
1813 00001CAA 8A4309            <1>       mov    al, [ebx+9] ; misc reg
1814 00001CAD EE                <1>       out    dx, al
1815                            <1>       ;
1816                            <1>       ; Enable video
1817 00001CAE 66BAC003          <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
1818 00001CB2 B020              <1>       mov    al, 20h
1819 00001CB4 EE                <1>         out     dx, al   ; set bit 5 to 1
1820 00001CB5 66BADA03          <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
1821 00001CB9 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 00001CBA F605[278E0100]80  <1>       test   byte [noclearmem], 80h
1828 00001CC1 7540              <1>       jnz    short _sm_15
1829                            <1>
1830                            <1>       ; 29/07/2016
1831 00001CC3 31C0              <1>       xor    eax, eax
1832 00001CC5 B900400000        <1>       mov    ecx, 4000h ; 16K words (32K)
1833 00001CCA 803D[3E8E0100]02  <1>       cmp     byte [VGA_MTYPE], 2  ; CTEXT, MTEXT, CGA
1834 00001CD1 7715              <1>       ja     short _sm_14    ; no ? (0A0000h)
1835 00001CD3 BF00800B00        <1>       mov    edi, 0B8000h
1836 00001CD8 7409              <1>       je     short _sm_13 ; CGA graphics mode
1837                            <1>       ; 08/08/2016
1838 00001CDA A3[3A8E0100]      <1>       mov    [VGA_INT43H], eax ; 0 ; default font
1839 00001CDF 66B82007          <1>       mov    ax, 0720h    ; CGA text mode
1840                            <1> _sm_13:
1841 00001CE3 F366AB            <1>       rep    stosw
1842 00001CE6 EB1B              <1>       jmp    short _sm_15
1843                            <1>
1844                            <1> _sm_14:
1845 00001CE8 BF00000A00        <1>       mov    edi, 0A0000h
1846                            <1>       ; ecx = 16384 dwords (64K)
1847 00001CED 66BAC403          <1>       mov    dx, 3C4h ; VGAREG_SEQU_ADDRESS
1848 00001CF1 B002              <1>       mov    al, 2
```

```
1849 00001CF3 EE                  <1>       out    dx, al
1850                              <1>       ;mov   dx, 3C5h ; VGAREG_SEQU_DATA
1851 00001CF4 6642                <1>       inc    dx
1852 00001CF6 EC                  <1>       in     al, dx ; mmask
1853 00001CF7 6650                <1>       push   ax
1854 00001CF9 B00F                <1>       mov    al, 0Fh ; all planes
1855 00001CFB EE                  <1>       out    dx, al
1856 00001CFC 30C0                <1>       xor    al, al ; 0
1857 00001CFE F3AB                <1>       rep    stosd  ; ecx = 163684 (64K)
1858 00001D00 6658                <1>       pop    ax
1859 00001D02 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 00001D03 668B03              <1>       mov    ax, [ebx] ; num of columns, 'twidth'
1864 00001D06 A2[BC6F0000]        <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 00001D0B FEC4                <1>       inc    ah ; 09/07/2016
1869 00001D0D 8825[C26F0000]      <1>       mov    [VGA_ROWS], ah
1870                              <1>       ; 10/08/2016
1871 00001D13 8A4302              <1>       mov    al, [ebx+2]
1872 00001D16 A2[BE6F0000]        <1>       mov    [CHAR_HEIGHT], al
1873                              <1>       ; 29/07/2016
1874                              <1>       ; length of regen buffer in bytes
1875 00001D1B 668B4B03            <1>       mov    cx, [ebx+3] ; 'slength_l'
1876 00001D1F 66890D[288E0100]    <1>       mov    [CRT_LEN], cx
1877                              <1>       ;
1878                              <1>       ; 27/07/2016
1879 00001D26 30E4                <1>       xor    ah, ah
1880 00001D28 A0[BE810100]        <1>       mov    al, [ACTIVE_PAGE] ; may be > 0 for mode 3
1881                              <1>       ;mul   word [CRT_LEN] ; 4096 for mode 3
1882 00001D2D 66F7E1              <1>       mul    cx ; 29/07/2016
1883 00001D30 66A3[AC810100]      <1>       mov    [CRT_START], ax
1884                              <1>       ;
1885 00001D36 B060                <1>       mov    al, 60h
1886                              <1>       ;cmp   byte [noclearmem], 0
1887                              <1>       ;jna   short _sm_16
1888                              <1>       ;add   al, 80h
1889 00001D38 0A05[278E0100]      <1>       or     al, [noclearmem] ; 17/11/2020
1890                              <1> _sm_16:
1891 00001D3E A2[BF6F0000]        <1>       mov    [VGA_VIDEO_CTL], al
1892 00001D43 C605[C06F0000]F9    <1>       mov    byte [VGA_SWITCHES], 0F9h
1893 00001D4A 8025[C16F0000]7F    <1>       and    byte [VGA_MODESET_CTL], 7Fh
1894                              <1>
1895 00001D51 5E                  <1>       pop    esi ; *
1896                              <1>
1897                              <1>       ; 26/07/2016
1898                              <1>       ; 07/07/2016
1899 00001D52 668B0D[D36F0000]    <1>       mov    cx, [CURSOR_MODE] ; restore cursor mode (initial value)
1900 00001D59 66870E              <1>       xchg   cx, [esi] ; cl = start line, ch = end line
1901                              <1>                        ; reset to initial value
1902 00001D5C 86E9                <1>       xchg   ch, cl  ; ch = start line, cl = end line
1903 00001D5E 66890D[D36F0000]    <1>       mov    [CURSOR_MODE], cx ; save (fixed) cursor mode
1904                              <1>
1905                              <1>       ; 27/07/2016
1906 00001D65 803D[3E8E0100]02    <1>       cmp    byte [VGA_MTYPE], 2 ; CTEXT, MTEXT
1907 00001D6C 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 00001D6E B40A                <1>       mov    ah, 10 ; 6845 register for cursor set
1915 00001D70 E84D060000          <1>       call   m16    ; output cx register
1916                              <1>
1917                              <1>       ; 25/07/2016
1918 00001D75 803D[BA6F0000]03    <1>         cmp     byte [CRT_MODE], 03h
1919 00001D7C 7507                <1>       jne    short _sm_17
1920                              <1>       ; 26/07/2016
1921                              <1>
1922 00001D7E A0[BE810100]        <1>       mov    al, [ACTIVE_PAGE]
1923 00001D83 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 00001D85 29C0                <1>       sub    eax, eax ; 17/11/2020
1928 00001D87 BF[AE810100]        <1>       mov    edi, CURSOR_POSN
1929 00001D8C AB                  <1>       stosd
1930 00001D8D AB                  <1>       stosd
1931 00001D8E AB                  <1>       stosd
1932 00001D8F 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 00001D90 803D[3E8E0100]02    <1>       cmp    byte [VGA_MTYPE], 2 ; CTEXT, MTEXT
1938 00001D97 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 00001D99 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 00001D9B E8DE160000          <1>       call   load_text_8_16_pat
1951                              <1>
1952                              <1>       ; video_func_1103h:
1953                              <1>       ; biosfn_set_text_block_specifier:
```

```
                                <1>       ; BL = font block selector code
                                <1>       ; NOTE: TRDOS 386 only uses and sets font block 0
                                <1>       ; (It is as BL = 0 for TRDOS 386)
00001DA0 66BAC403               <1>       mov    dx, 3C4h ; VGAREG_SEQU_ADDRESS
                                <1>       ;;mov  ah, bl
                                <1>       ;sub   ah, ah ; 0
                                <1>       ;mov   al, 03h
                                <1>       ; 19/11/2020
00001DA4 66B80300               <1>       mov    ax, 03h
00001DA8 66EF                   <1>       out    dx, ax
                                <1> _sm_19:
                                <1>       ; 29/07/2016
                                <1>       ; 26/07/2016
                                <1>       ; 24/06/2016
                                <1>       ;mov   edi, 0B8000h
                                <1>       ;mov   cx, 4000h ; 16K words (32K)
                                <1>       ;
00001DAA 30C0                   <1>       xor    al, al
00001DAC 3805[258E0100]         <1>         cmp     [p_crt_mode], al ; 0
00001DB2 7705                   <1>         ja      short _sm_20 ; 03h, 80h or 83h
                                <1>
                                <1>       ; case 1 - 19/11/2020
                                <1>       ;
                                <1>       ; If [pc_crt_mode] = 0, that means, previous mode is 03h
                                <1>       ; and current mode is not 03h (case 1)
                                <1>
                                <1>       ; 30/07/2016
                                <1>       ; 24/06/2016
                                <1>       ; TRDOS 386 (TRDOS v2) 'set mode' modification
                                <1>       ; (for multiscreen feature):
                                <1>       ; If '_set_mode' procedure is called for video mode 3
                                <1>       ;     while video mode is 3, video page will be cleared
                                <1>       ;     and cursor position of video page will be reset.
                                <1>       ; If '_set_mode' procedure is called for a video mode
                                <1>       ;     except video mode 3, while current video mode
                                <1>       ;     is 3, all video pages of mode 3 will be copied
                                <1>       ;     to 98000h address as backup, before mode change.
                                <1>       ; If '_set_mode' procedure is called for video mode 3
                                <1>       ;     while video mode is not 3 and if there is video
                                <1>       ;     page backup for video mode 3, all (of 8) mode 3
                                <1>       ;     video pages will be restored from 98000h.
                                <1>
                                <1>       ; 19/11/2020
                                <1>       ;mov   [ACTIVE_PAGE], al ; 0
                                <1>
                                <1>       ; Here,
                                <1>       ; video memory already cleared if [noclearmem] <> 80h
                                <1>
                                <1>       ;mov   ax, 0720h
                                <1>       ;mov   cx, 4000h ; 16K words (32K)
                                <1>       ;mov   edi, 0B8000h
                                <1>       ;rep   stosw
                                <1>       ;sub   al, al
                                <1>
                                <1>       ;jmp   short _sm_23
                                <1>
                                <1>       ; Set hardware side for the new active video page
                                <1>
00001DB4 E9FB010000             <1>       jmp    _set_active_page ; 19/11/2020
                                <1>
                                <1> _sm_20:
                                <1>       ; 19/11/2020
                                <1>       ; case 2 or case 3 or case 4 - 19/11/2020
                                <1>
                                <1>       ; 19/11/2020
00001DB9 803D[BA6F0000]03       <1>       cmp    byte [CRT_MODE], 3  ; new video mode
00001DC0 754F                   <1>       jne    short _sm_22 ; al = 0 (& video mode <> 03h)
                                <1>                           ; case 4 - 19/11/2020
                                <1>                           ; ([p_crt_mode] = 83h)
                                <1>
                                <1>       ; case 2 or case 3 - 19/11/2020
                                <1>
                                <1>       ;movzx ebx, byte [ACTIVE_PAGE]
                                <1>       ; 19/11/2020
00001DC2 A0[268E0100]           <1>       mov    al, [p_crt_page] ; previous mode 3 active page
                                <1>       ;movzx ebx, al
                                <1>       ;shl   bl, 1 ; * 2
                                <1>       ;add   ebx, CURSOR_POSN
                                <1>
                                <1>       ; 29/07/2016
00001DC7 F605[258E0100]7F       <1>       test    byte [p_crt_mode], 7Fh ; 83h or 80h or 03h
00001DCE 740F                   <1>       jz     short _sm_21 ; do not restore video pages
                                <1>                           ; case 2 - 19/11/2020
                                <1>       ; case 3 - 19/11/2020
                                <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>       ; 19/11/2020
00001DD0 A2[BE810100]           <1>       mov    [ACTIVE_PAGE], al ; current mode 3 active page
                                <1>
                                <1>       ; 12/12/2020
                                <1>       ;; restore video pages
                                <1>       ;mov   esi, 98000h ; 30/07/2016
                                <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
00001DD5 E89C000000             <1>       call   _restore_mode3_multiscreen
                                <1>
                                <1>       ;jmp   short _sm_23 ; do not clear current video pages
                                <1>
                                <1>       ; 19/11/2020
00001DDA 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
00001DDF F605[278E0100]80       <1>       test   byte [noclearmem], 80h
00001DE6 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
00001DE8 66B82007               <1>       mov    ax, 0720h
                                <1>       ; 26/11/2020
00001DEC B900080000             <1>       mov    ecx, 2048 ; 4096/2
00001DF1 BF00800B00             <1>       mov    edi, 0B8000h ; [crt_base]
00001DF6 66033D[AC810100]       <1>       add    di, [CRT_START]
00001DFD F366AB                 <1>       rep    stosw  ; FILL THE REGEN BUFFER WITH BLANKS
                                <1>       ;
                                <1>       ; 19/11/2020
00001E00 A0[BE810100]           <1>       mov    al, [ACTIVE_PAGE] ; 0 to 7 (for video mode 3)
00001E05 0FB6D8                 <1>       movzx  ebx, al
00001E08 D0E3                   <1>       shl    bl, 1
00001E0A 66898B[AE810100]       <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
00001E11 8025[258E0100]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
00001E18 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
00001E1D BE00800B00             <1>       mov    esi, 0B8000h
00001E22 BF00800900             <1>       mov    edi, 98000h ; 30/07/2016
00001E27 B900200000             <1>       mov    ecx, (0B8000h-0B0000h)/4
00001E2C F3A5                   <1>       rep    movsd
                                <1>
00001E2E C605[258E0100]03       <1>       mov    byte [p_crt_mode], 3 ; previous mode, backup sign
                                <1>       ; 26/11/2020
00001E35 860D[BE810100]         <1>       xchg   cl, [ACTIVE_PAGE]
00001E3B 880D[268E0100]         <1>       mov    [p_crt_page], cl  ; save as previous active page
                                <1>
                                <1>       ; save cursor positions
00001E41 BE[AE810100]           <1>       mov    esi, CURSOR_POSN
00001E46 BF[2A8E0100]           <1>       mov    edi, cursor_pposn ; cursor positions backup
00001E4B B104                   <1>       mov    cl, 4
00001E4D F3A5                   <1>       rep    movsd
00001E4F 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
                                <1>
00001E50 A0[268E0100]           <1>       mov    al, [p_crt_page] ; previous mode 3 active page
00001E55 A2[BE810100]           <1>       mov    [ACTIVE_PAGE], al ; current mode 3 active page
                                <1>
                                <1>       ; 12/12/2020
                                <1>       ; moved here from 'vesa_vbe3_pmi'
                                <1>
                                <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 00001E5A 66C705[AC810100]00-    <1>       mov    word [CRT_START], 0
2167 00001E62 00                     <1>
2168                                 <1>
2169                                 <1>       ; check active page and set it again if it is not 0
2170 00001E63 08C0                   <1>       or     al, al
2171                                 <1>       ;;jz   short vbe3_pmi_7
2172                                 <1>       ;jz    short _restore_mode3_multiscreen
2173 00001E65 740F                   <1>       jz     short r_m3_ms_1
2174 00001E67 88C1                   <1>       mov    cl, al
2175                                 <1> ;vbe3_pmi_5:
2176                                 <1> r_m3_ms_0:
2177 00001E69 668105[AC810100]00-    <1>       add    word [CRT_START], 4096
2177 00001E71 10                     <1>
2178 00001E72 FEC9                   <1>       dec    cl
2179                                 <1>       ;jnz   short vbe3_pmi_5
2180 00001E74 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 00001E76 BE00800900             <1>       mov    esi, 98000h ; 30/07/2016
2189 00001E7B BF00800B00             <1>       mov    edi, 0B8000h
2190                                 <1>       ;mov   cx, 2000h ; 8K dwords (32K)
2191 00001E80 B900200000             <1>       mov    ecx, 2000h
2192 00001E85 F3A5                   <1>       rep    movsd
2193                                 <1>
2194                                 <1>       ; 19/11/2020
2195 00001E87 880D[258E0100]         <1>       mov    [p_crt_mode], cl ; reset ('case 3' end condition)
2196                                 <1>
2197                                 <1>       ; restore cursor positions
2198 00001E8D BE[2A8E0100]           <1>       mov    esi, cursor_pposn
2199 00001E92 BF[AE810100]           <1>       mov    edi, CURSOR_POSN
2200                                 <1>       ;mov   ecx, 4 ; restore all cursor positions (16 bytes)
2201 00001E97 B104                   <1>       mov    cl, 4
2202 00001E99 F3A5                   <1>       rep    movsd
2203 00001E9B 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 00001E9C 803D[BE6F0000]08       <1>       cmp    byte [CHAR_HEIGHT], 8
2227 00001EA3 7649                   <1>       jna    short csf_3
2228 00001EA5 80FC08                 <1>       cmp    ah, 8
2229 00001EA8 7344                   <1>       jnb    short csf_3
2230 00001EAA 3C20                   <1>       cmp    al, 20h
2231 00001EAC 7340                   <1>       jnb    short csf_3
2232                                 <1>       ;
2233 00001EAE 6650                   <1>       push   ax
2234                                 <1>       ; {
2235                                 <1>       ; if(CL!=(CH+1))
2236 00001EB0 FEC0                   <1>       inc    al
2237 00001EB2 38C4                   <1>       cmp    ah, al   ; ah != al + 1
2238 00001EB4 740F                   <1>         je      short csf_1
2239                                 <1>       ; CH = ((CH+1) * cheight / 8) -1;
2240 00001EB6 8A25[BE6F0000]         <1>       mov    ah, [CHAR_HEIGHT]
2241 00001EBC F6E4                   <1>       mul    ah
2242 00001EBE C0E803                 <1>       shr    al, 3 ; / 8
2243 00001EC1 FEC8                   <1>       dec    al ; - 1
2244 00001EC3 EB0E                   <1>       jmp    short csf_2
2245                                 <1> csf_1:
2246                                 <1>       ; }
2247                                 <1>       ; else       ; ah = al + 1
2248                                 <1>       ; {
2249 00001EC5 FEC4                   <1>       inc    ah     ; ah = ah + 1
2250                                 <1>       ; CH = ((CL+1) * cheight / 8) - 2;
2251 00001EC7 A0[BE6F0000]           <1>       mov    al, [CHAR_HEIGHT]
2252 00001ECC F6E4                   <1>       mul    ah
2253 00001ECE C0E803                 <1>       shr    al, 3 ; / 8
2254 00001ED1 2C02                   <1>       sub    al, 2 ; - 2
2255                                 <1>       ; al = 14 (if [CHAR_HEIGHT] = 16)
2256                                 <1> csf_2:
2257 00001ED3 880424                 <1>       mov    [esp], al
2258 00001ED6 8A642401               <1>       mov    ah, [esp+1]
2259                                 <1>       ; CL = ((CL+1) * cheight / 8) - 1;
2260 00001EDA FEC4                   <1>       inc    ah
2261 00001EDC A0[BE6F0000]           <1>       mov    al, [CHAR_HEIGHT]
2262 00001EE1 F6E4                   <1>       mul    ah
2263 00001EE3 C0E803                 <1>       shr    al, 3 ; / 8
2264 00001EE6 FEC8                   <1>       dec    al ; - 1
2265 00001EE8 88442401               <1>       mov    [esp+1], al
2266                                 <1>       ; ah = 15 (if [CHAR_HEIGHT] = 16)
```

```
                                <1>       ;
00001EEC 6658                   <1>       pop    ax
                                <1> csf_3:
00001EEE C3                     <1>       retn
                                <1>
                                <1> SET_CTYPE:
                                <1>       ; 12/09/2016
                                <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
00001EEF 803D[BA6F0000]07       <1>       cmp    byte [CRT_MODE], 7
00001EF6 0F875BFCFFFF           <1>       ja     VIDEO_RETURN ; 12/09/2016
00001EFC E805000000             <1>       call   _set_ctype
00001F01 E951FCFFFF             <1>         jmp     VIDEO_RETURN
                                <1>
                                <1> _set_ctype:
                                <1>       ; 02/09/2014 (Retro UNIX 386 v1)
                                <1>       ;
                                <1>       ; VIDEO.ASM - 06/10/85  VIDEO DISPLAY BIOS
                                <1>
                                <1>       ; (CH) = BITS 4-0 = START LINE FOR CURSOR
                                <1>       ;  ** HARDWARE WILL ALWAYS CAUSE BLINK
                                <1>       ;  ** SETTING BIT 5 OR 6 WILL CAUSE ERRATIC BLINKING
                                <1>       ;     OR NO CURSOR AT ALL
                                <1>       ; (CL) = BITS 4-0 = END LINE FOR CURSOR
                                <1>
                                <1> ;--------------------------------------------
                                <1> ; SET_CTYPE
                                <1> ;     THIS ROUTINE SETS THE CURSOR VALUE
                                <1> ; INPUT
                                <1> ;     (CX) HAS CURSOR VALUE CH-START LINE, CL-STOP LINE
                                <1> ; OUTPUT
                                <1> ;     NONE
                                <1> ;--------------------------------------------
                                <1>
                                <1>       ; 07/07/2016
                                <1>       ; Fixing cursor start and stop line depending on
                                <1>       ; current character height (=16)
                                <1>       ; (Note: Default/initial values are 6 and 7.
                                <1>       ; If set values are 6 (start) & 7 (stop) and
                                <1>       ; [CHAR_HEIGHT] = 16 :
                                <1>       ; After fixing, start line will be 14, stop line
                                <1>       ; will be 15.)
00001F06 6689C8                 <1>       mov    ax, cx
00001F09 86C4                   <1>       xchg   al, ah
                                <1>       ; AL = start line, AH = stop line
00001F0B E88CFFFFFF             <1>       call   cursor_shape_fix
                                <1>       ; AL = start line (fixed), AH = stop line (fixed)
00001F10 6689C1                 <1>       mov    cx, ax
00001F13 86E9                   <1>       xchg   ch, cl
                                <1>       ; CH = start line (fixed), CL = stop line (fixed)
                                <1>       ;
00001F15 B40A                   <1>       mov    ah, 10 ; 6845 register for cursor set
00001F17 66890D[D36F0000]       <1>       mov    [CURSOR_MODE], cx ; save in data area
                                <1>       ;call  m16    ; output cx register
                                <1>       ;retn
00001F1E E99F040000             <1>         jmp     m16
                                <1>
                                <1> SET_CPOS:
                                <1>       ; 12/09/2016
                                <1>       ; 07/07/2016
                                <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
00001F23 80FF07                 <1>       cmp    bh, 7 ; video page > 7 ; 07/07/2016
00001F26 0F872BFCFFFF           <1>       ja     VIDEO_RETURN
                                <1>       ;
00001F2C 803D[BA6F0000]07       <1>       cmp    byte [CRT_MODE], 7
00001F33 770A                   <1>       ja     short vga_set_cpos ; 12/09/2016
00001F35 E85D040000             <1>       call   _set_cpos
00001F3A E918FCFFFF             <1>         jmp     VIDEO_RETURN
                                <1>
                                <1> vga_set_cpos:
                                <1>       ; 12/09/2016
                                <1>       ; 09/07/2016
                                <1>       ; set cursor position
                                <1>       ; NOTE: Hardware cursor position will not be set
                                <1>       ;   in any VGA modes (>7)
                                <1>       ;   But, cursor position will be saved into
                                <1>       ;   [CURSOR_POSN].
                                <1>       ;   TRDOS 386 (TRDOS v2.0) uses only one page
                                <1>       ;   (page 0) for all graphics modes.
                                <1>
00001F3F 668915[AE810100]       <1>       mov    [CURSOR_POSN], dx ; save cursor pos for pg 0
                                <1>       ; 04/08/2016
                                <1>       ;mov   bh, [ACTIVE_PAGE] ; = 0
                                <1>       ;call  _set_cpos
00001F46 E90CFCFFFF             <1>       jmp     VIDEO_RETURN
                                <1>
                                <1> READ_CURSOR:
                                <1>       ; 12/09/2016
                                <1>       ; 07/07/2016
                                <1>       ; 12/05/2016
                                <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>       ;
                                <1>       ; VIDEO.ASM - 06/10/85  VIDEO DISPLAY BIOS
                                <1>
                                <1> ;--------------------------------------------------------
                                <1> ; READ_CURSOR
                                <1> ;     THIS ROUTINE READS THE CURRENT CURSOR VALUE FROM THE
                                <1> ;     845, FORMATS IT, AND SENDS IT BACK TO THE CALLER
                                <1> ; INPUT
                                <1> ;     BH - PAGE OF CURSOR
                                <1> ; OUTPUT
                                <1> ;     DX - ROW, COLUMN OF THE CURRENT CURSOR POSITION
                                <1> ;     CX - CURRENT CURSOR MODE
                                <1> ;--------------------------------------------------------
                                <1>
                                <1>       ; BH = Video page number (0 to 7)
```

```
                                <1>
                                <1>       ; 07/07/2016
00001F4B 80FF07                 <1>       cmp    bh, 7 ; video page > 7 (invalid)
00001F4E 7606                   <1>       jna    short read_cursor_1
                                <1>       ; invalid video page (input)
00001F50 31C9                   <1>       xor    ecx, ecx ; 0
00001F52 31D2                   <1>       xor    edx, edx ; 0
00001F54 EB15                   <1>       jmp    short read_cursor_2
                                <1> read_cursor_1:
                                <1>       ; 12/09/2016
00001F56 803D[BA6F0000]07       <1>       cmp    byte [CRT_MODE], 7 ; vga mode
00001F5D 7727                   <1>       ja     short vga_get_cpos
                                <1>       ;
00001F5F E815000000             <1>       call   get_cpos
00001F64 0FB70D[D36F0000]       <1>       movzx  ecx, word [CURSOR_MODE]
                                <1> read_cursor_2:
00001F6B 5D                     <1>       pop    ebp
00001F6C 5F                     <1>       pop    edi
00001F6D 5E                     <1>       pop    esi
00001F6E 5B                     <1>       pop    ebx
00001F6F 58                     <1>       pop    eax  ; DISCARD SAVED CX AND DX
00001F70 58                     <1>       pop    eax
00001F71 A1[188E0100]           <1>       mov    eax, [video_eax] ; 12/05/2016
                                <1>       ;;15/01/2017
                                <1>       ;;mov  byte [intflg], 0 ; 07/01/2017
00001F76 1F                     <1>       pop    ds
00001F77 07                     <1>       pop    es
00001F78 CF                     <1>       iretd
                                <1>
                                <1> get_cpos:
                                <1>       ; 12/05/2016
                                <1>       ; 16/01/2016
                                <1>       ; BH = Video page number (0 to 7)
                                <1>       ;
00001F79 D0E7                   <1>       shl    bh, 1 ; WORD OFFSET
00001F7B 0FB6F7                 <1>       movzx  esi, bh
00001F7E 0FB796[AE810100]       <1>       movzx  edx, word [esi+CURSOR_POSN]
00001F85 C3                     <1>       retn
                                <1>
                                <1> vga_get_cpos:
                                <1>       ; 12/09/2016
                                <1>       ; get cursor position (vga)
00001F86 0FB715[AE810100]       <1>       movzx  edx, word [CURSOR_POSN] ; cursor pos for pg 0
00001F8D 31C9                   <1>       xor    ecx, ecx ; Cursor Mode = 0 (invalid)
00001F8F EBDA                   <1>       jmp     short read_cursor_2
                                <1>
                                <1> ACT_DISP_PAGE:
                                <1>       ; 07/07/2016
                                <1>       ; 26/06/2016
                                <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>       ;
                                <1>       ; VIDEO.ASM - 06/10/85  VIDEO DISPLAY BIOS
                                <1>       ;
                                <1> ;--------------------------------------------------
                                <1> ; ACT_DISP_PAGE
                                <1> ;     THIS ROUTINE SETS THE ACTIVE DISPLAY PAGE, ALLOWING
                                <1> ;     THE FULL USE OF THE MEMORY SET ASIDE FOR THE VIDEO ATTACHMENT
                                <1> ; INPUT
                                <1> ;     AL HAS THE NEW ACTIVE DISPLAY PAGE
                                <1> ; OUTPUT
                                <1> ;     THE 6845 IS RESET TO DISPLAY THAT PAGE
                                <1> ;--------------------------------------------------
                                <1>       ; 07/07/2016
00001F91 3C07                   <1>       cmp    al, 7  ; > 7 = invalid video page number
                                <1>       ;ja    VIDEO_RETURN
00001F93 7715                   <1>         ja   short adp_2 ; 18/11/2020
                                <1>       ;cmp   byte [CRT_MODE], 3
                                <1>       ;je    short adp_1
                                <1>       ; 18/11/2020
00001F95 8A25[BA6F0000]         <1>       mov    ah, [CRT_MODE]
00001F9B 80FC03                 <1>       cmp    ah, 3
00001F9E 7605                   <1>       jna    short adp_1 ; mode 01h, 00h (01h), 02h (03h), 03h
00001FA0 80FC07                 <1>       cmp    ah, 7      ; mode 07h (03h)
00001FA3 7505                   <1>       jne    short adp_2
                                <1>       ;and   al, al
                                <1>         ;jnz VIDEO_RETURN
                                <1>       ;;sub  al, al ; 0 ; force to page 0
                                <1> adp_1:
00001FA5 E805000000             <1>       call   set_active_page
                                <1> adp_2:
00001FAA E9A8FBFFFF             <1>         jmp     VIDEO_RETURN
                                <1>
                                <1> set_active_page:   ; tty_sw
                                <1>       ; 09/12/2017
                                <1>       ; 26/07/2016
                                <1>       ; 26/06/2016
                                <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>       ; 30/06/2015
                                <1>       ; 04/03/2014  (act_disp_page --> tty_sw)
                                <1>       ; 10/12/2013
                                <1>       ; 04/12/2013
                                <1>       ;
00001FAF A2[BE810100]           <1>       mov    [ACTIVE_PAGE], al ; save active page value ; [ptty]
                                <1> _set_active_page:
                                <1>       ; 27/06/2015
00001FB4 0FB6D8                 <1>       movzx  ebx, al
                                <1>       ;
                                <1>       ;cbw   ; 07/09/2014 (ah=0)
00001FB7 28E4                   <1>       sub    ah, ah ; 09/12/2017
00001FB9 66F725[288E0100]       <1>       mul    word [CRT_LEN]  ; get saved length of regen buffer
                                <1>                              ; display page times regen length
                                <1>       ; 10/12/2013
00001FC0 66A3[AC810100]         <1>       mov    [CRT_START], ax ; save start address for later
00001FC6 6689C1                 <1>       mov    cx, ax ; start address to cx
                                <1> _M16:
```

```
                                <1>       ;sar   cx, 1
00001FC9 66D1E9                 <1>       shr    cx, 1  ; divide by 2 for 6845 handling
00001FCC B40C                   <1>       mov    ah, 12 ; 6845 register for start address
00001FCE E8EF030000             <1>       call   m16
                                <1>       ;sal   bx, 1
                                <1>       ; 01/09/2014
00001FD3 D0E3                   <1>       shl    bl, 1  ; *2 for word offset
00001FD5 81C3[AE810100]         <1>       add    ebx, CURSOR_POSN
00001FDB 668B13                 <1>       mov    dx, [ebx] ; get cursor for this page
                                <1>       ; 16/01/2016
                                <1>       ;call  m18
                                <1>       ;retn
00001FDE 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
00001FE3 0FB605[BC6F0000]       <1>       movzx  eax, byte [CRT_COLS] ; 24/06/2016
00001FEA F6E6                   <1>       mul    dh      ; row value
00001FEC 30F6                   <1>       xor    dh, dh   ; 0
00001FEE 6601D0                 <1>       add    ax, dx  ; add column value to the result
00001FF1 66D1E0                 <1>       shl    ax, 1  ; * 2 for attribute bytes
                                <1>              ; EAX = AX = OFFSET OF CHAR POSITION IN REGEN BUFFER
00001FF4 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>
00001FF5 0FB6CF                 <1>       movzx  ecx, bh ; video page number
00001FF8 89CE                   <1>       mov    esi, ecx
00001FFA 66D1E6                 <1>       shl    si, 1
00001FFD 668B96[AE810100]       <1>       mov    dx, [esi+CURSOR_POSN]
00002004 740C                   <1>       jz     short p21
00002006 6631F6                 <1>       xor    si, si
                                <1> p20:
00002009 660335[288E0100]       <1>       add    si, [CRT_LEN] ; 24/06/2016
                                <1>       ;add   si, 80*25*2 ; add length of buffer for one page
00002010 E2F7                   <1>       loop   p20
                                <1> p21:
00002012 6621D2                 <1>       and    dx, dx
00002015 7407                   <1>       jz     short p22
00002017 E8C7FFFFFF             <1>       call   position ; determine location in regen in page
0000201C 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
0000201E 66BADA03               <1>       mov    dx, 03DAh ; status port
                                <1>       ; cx = 0
00002022 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>
```

```
2582                                <1>       ; 07/07/2016
2583 00002023 38F5                  <1>       cmp    ch, dh
2584 00002025 0F872CFBFFFF          <1>       ja     VIDEO_RETURN
2585 0000202B 38D1                  <1>       cmp    cl, dl
2586 0000202D 0F8724FBFFFF          <1>       ja     VIDEO_RETURN
2587                                <1>       ;
2588 00002033 E805000000            <1>       call   _scroll_up
2589 00002038 E91AFBFFFF            <1>         jmp     VIDEO_RETURN
2590                                <1>
2591                                <1> _scroll_up:  ; from 'write_tty'
2592                                <1>       ;
2593                                <1>       ; cl = left upper column
2594                                <1>       ; ch = left upper row
2595                                <1>       ; dl = right lower column
2596                                <1>       ; dh = right lower row
2597                                <1>       ;
2598                                <1>       ; al = line count
2599                                <1>       ; bl = attribute to be used on blanked line
2600                                <1>       ; bh = video page number (0 to 7)
2601                                <1>
2602 0000203D E89C000000            <1>       call   test_line_count ; 16/01/2016
2603                                <1>
2604 00002042 8A25[BA6F0000]        <1>       mov    ah, [CRT_MODE] ; current video mode
2605                                <1>       ;;cmp  byte [CRT_MODE], 4
2606                                <1>       ;cmp   ah, 4 ; 07/07/2016
2607                                <1>       ;jnb   GRAPHICS_UP ; 26/06/2016
2608                                <1>       ; 18/11/2020
2609 00002048 80FC04                <1>       cmp    ah, 4
2610 0000204B 720A                  <1>       jb     short n0
2611 0000204D 80FC07                <1>       cmp    ah, 7 ; TEST FOR BW CARD
2612                                <1>                    ;      (80x25 text, mono)
2613 00002050 7405                  <1>       je     short n0 ; same with mode 3 for TRDOS 386
2614 00002052 E938050000            <1>       jmp    GRAPHICS_UP
2615                                <1> n0:
2616                                <1>       ; 07/07/2016
2617 00002057 80FF07                <1>       cmp    bh, 7 ; video page number
2618 0000205A 7606                  <1>       jna    short n1
2619 0000205C 8A3D[BE810100]        <1>       mov    bh, [ACTIVE_PAGE]
2620                                <1> n1:
2621 00002062 88DC                  <1>       mov    ah, bl ; attribute
2622 00002064 6650                  <1>       push   ax ; *
2623                                <1>       ;mov   esi, [CRT_BASE]
2624 00002066 BE00800B00            <1>         mov     esi, 0B8000h
2625 0000206B 3A3D[BE810100]        <1>         cmp     bh, [ACTIVE_PAGE]
2626 00002071 750B                  <1>       jne    short n2
2627                                <1>       ;
2628 00002073 66A1[AC810100]        <1>         mov     ax, [CRT_START]
2629 00002079 6601C6                <1>         add     si, ax
2630 0000207C EB11                  <1>         jmp     short n4
2631                                <1> n2:
2632 0000207E 20FF                  <1>         and     bh, bh
2633 00002080 740D                  <1>       jz     short n4
2634 00002082 88F8                  <1>       mov    al, bh
2635                                <1> n3:
2636 00002084 660335[288E0100]      <1>         add si, [CRT_LEN]
2637 0000208B FEC8                  <1>         dec al
2638 0000208D 75F5                  <1>       jnz    short n3
2639                                <1> n4:
2640 0000208F E85D000000            <1>       call   scroll_position ; 16/01/2016
2641 00002094 7420                  <1>         jz      short n6
2642                                <1>
2643 00002096 01CE                  <1>         add     esi, ecx ; from address for scroll
2644 00002098 88F5                  <1>       mov    ch, dh  ; #rows in block
2645 0000209A 28C5                  <1>       sub    ch, al ; #rows to be moved
2646                                <1> n5:
2647 0000209C E894000000            <1>       call   n10 ; 16/01/2016
2648                                <1>
2649 000020A1 51                    <1>         push ecx
2650 000020A2 0FB60D[BC6F0000]      <1>       movzx  ecx, byte [CRT_COLS]
2651 000020A9 00C9                  <1>       add    cl, cl
2652 000020AB 01CE                  <1>         add  esi, ecx  ; next line
2653 000020AD 01CF                  <1>         add  edi, ecx
2654 000020AF 59                    <1>       pop    ecx
2655                                <1>
2656 000020B0 FECD                  <1>       dec    ch      ; count of lines to move
2657 000020B2 75E8                  <1>       jnz    short n5 ; row loop
2658                                <1>       ; ch = 0
2659 000020B4 88C6                  <1>       mov    dh, al  ; #rows
2660                                <1> n6:
2661                                <1>       ; attribute in ah
2662 000020B6 B020                  <1>       mov    al, ' '       ; fill with blanks
2663                                <1> n7:
2664 000020B8 E885000000            <1>       call   n11 ; 16/01/2016
2665                                <1>
2666 000020BD 8A0D[BC6F0000]        <1>       mov    cl, [CRT_COLS]
2667 000020C3 00C9                  <1>       add    cl, cl
2668 000020C5 01CF                  <1>         add  edi, ecx
2669                                <1>
2670 000020C7 FECE                  <1>       dec    dh
2671 000020C9 75ED                  <1>       jnz    short n7
2672                                <1> n16:
2673 000020CB 3A3D[BE810100]        <1>       cmp    bh, [ACTIVE_PAGE]
2674 000020D1 750A                  <1>       jne    short n8
2675                                <1>
2676                                <1>       ;cmp   byte [CRT_MODE], 7 ; is this the black and white card
2677                                <1>       ;je    short n8        ; if so, skip the mode reset
2678                                <1>
2679 000020D3 A0[BB6F0000]          <1>       mov    al, [CRT_MODE_SET] ; get the value of mode set
2680 000020D8 66BAD803              <1>       mov    dx, 03D8h ; always set color card port
2681 000020DC EE                    <1>       out    dx, al
2682                                <1> n8:
2683 000020DD C3                    <1>       retn
2684                                <1>
2685                                <1> test_line_count:
2686                                <1>       ; 12/05/2016
```

```
                        <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
                        <1>       ; 07/09/2014 (scroll_up)
000020DE 08C0           <1>       or    al, al
000020E0 740E           <1>       jz    short al_set2
000020E2 6652           <1>       push  dx
000020E4 28EE           <1>       sub   dh, ch  ; subtract upper row from lower row number
000020E6 FEC6           <1>       inc   dh    ; adjust difference by 1
000020E8 38C6           <1>       cmp   dh, al       ; line count = amount of rows in window?
000020EA 7502           <1>       jne   short al_set1 ; if not the we're all set
000020EC 30C0           <1>       xor   al, al ; otherwise set al to zero
                        <1> al_set1:
000020EE 665A           <1>       pop   dx
                        <1> al_set2:
000020F0 C3             <1>       retn
                        <1>
                        <1> scroll_position:
                        <1>       ; 26/06/2016
                        <1>       ; 30/01/2016
                        <1>         ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
                        <1>       ; 07/09/2014 (scroll_up)
                        <1>
000020F1 6652           <1>       push  dx
000020F3 6689CA         <1>       mov   dx, cx ; now, upper left position in DX
000020F6 E8E8FEFFFF     <1>       call  position
000020FB 01C6           <1>       add   esi, eax
000020FD 89F7           <1>       mov   edi, esi
000020FF 665A           <1>       pop   dx    ; lower right position in DX
00002101 6629CA         <1>       sub   dx, cx
00002104 FEC6           <1>       inc   dh    ; dh = #rows
00002106 FEC2           <1>       inc   dl    ; dl = #cols in block
00002108 59             <1>       pop   ecx   ; return address
00002109 6658           <1>       pop   ax    ; * ; al = line count, ah = attribute
0000210B 51             <1>       push  ecx   ; return address
0000210C 0FB7C8         <1>       movzx ecx, ax
0000210F 8A25[BC6F0000] <1>       mov   ah, [CRT_COLS]
00002115 F6E4           <1>       mul   ah    ; determine offset to from address
00002117 6601C0         <1>       add   ax, ax  ; *2 for attribute byte
                        <1>       ;
0000211A 6650           <1>       push  ax    ; offset
0000211C 6652           <1>       push  dx
                        <1>       ;
                        <1>       ; 04/04/2014
0000211E 66BADA03       <1>       mov   dx, 3DAh ; guaranteed to be color card here
                        <1> n9:                    ; wait_display_enable
00002122 EC             <1>         in      al, dx   ; get port
00002123 A808           <1>       test  al, RVRT ; wait for vertical retrace
00002125 74FB           <1>       jz    short n9 ; wait_display_enable
00002127 B025           <1>       mov   al, 25h
00002129 B2D8           <1>       mov   dl, 0D8h ; address control port
0000212B EE             <1>       out   dx, al ; turn off video during vertical retrace
0000212C 665A           <1>       pop   dx    ; #rows, #cols
0000212E 6658           <1>             pop   ax    ; offset
00002130 6691           <1>       xchg  ax, cx ;
                        <1>       ; ecx = offset, al = line count, ah = attribute
                        <1>       ;
00002132 08C0           <1>       or    al, al
00002134 C3             <1>       retn
                        <1> n10:
                        <1>       ; Move rows
00002135 88D1           <1>       mov   cl, dl ; get # of cols to move
00002137 56             <1>       push  esi
00002138 57             <1>       push  edi   ; save start address
                        <1> n10r:
00002139 66A5           <1>       movsw       ; move that line on screen
0000213B FEC9           <1>       dec   cl
0000213D 75FA           <1>         jnz     short n10r
0000213F 5F             <1>       pop   edi
00002140 5E             <1>       pop   esi   ; recover addresses
00002141 C3             <1>       retn
                        <1> n11:
                        <1>       ; Clear rows
                        <1>                   ; dh =  #rows
00002142 88D1           <1>         mov  cl, dl ; get # of cols to clear
00002144 57             <1>         push    edi     ; save address
                        <1> n11r:
00002145 66AB           <1>         stosw           ; store fill character
00002147 FEC9           <1>       dec   cl
00002149 75FA           <1>         jnz     short n11r
0000214B 5F             <1>         pop     edi     ; recover address
0000214C C3             <1>       retn
                        <1>
                        <1> SCROLL_DOWN:
                        <1>       ; 07/07/2016
                        <1>       ; 27/06/2016
                        <1>       ; 26/06/2016
                        <1>       ; 12/05/2016
                        <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
                        <1>       ;
                        <1>       ; VIDEO.ASM - 06/10/85  VIDEO DISPLAY BIOS
                        <1>
                        <1> ;--------------------------------------------
                        <1> ; SCROLL DOWN
                        <1> ;     THIS ROUTINE MOVES THE CHARACTERS WITHIN A DEFINED
                        <1> ;     BLOCK DOWN ON THE SCREEN, FILLING THE TOP LINES
                        <1> ;     WITH A DEFINED CHARACTER
                        <1> ; INPUT
                        <1> ;     (AH) = CURRENT CRT MODE
                        <1> ;     (AL) = NUMBER OF LINES TO SCROLL
                        <1> ;     (CX) = UPPER LEFT CORNER OF RECION
                        <1> ;     (DX) = LOWER RIGHT CORNER OF REGION
                        <1> ;     (BH) = FILL CHARACTER
                        <1> ;     (DS) = DATA SEGMENT
                        <1> ;     (ES) = REGEN SEGMENT
                        <1> ; OUTPUT
                        <1> ;     NONE -- SCREEN IS SCROLLED
```

```
                        <1> ;-----------------------------------------
                        <1>
                        <1>       ; 07/07/2016
0000214D 38F5           <1>       cmp    ch, dh
                        <1>       ;ja    VIDEO_RETURN
0000214F 7709           <1>       ja     short _s_d_retn ; 18/11/2020
00002151 38D1           <1>       cmp    cl, dl
                        <1>       ;ja    VIDEO_RETURN
00002153 7705           <1>       ja     short _s_d_retn ; 18/11/2020
                        <1>       ;
00002155 E805000000     <1>       call   _scroll_down
                        <1> _s_d_retn:
0000215A 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>       ; !!!!
0000215F FD             <1>       std          ; DIRECTION FOR SCROLL DOWN
                        <1>       ; !!!!
00002160 E879FFFFFF     <1>       call   test_line_count ; 16/01/2016
                        <1>
00002165 8A25[BA6F0000] <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
0000216B 80FC04         <1>       cmp    ah, 4
0000216E 720A           <1>       jb     short _n0
00002170 80FC07         <1>       cmp    ah, 7 ; TEST FOR BW CARD
                        <1>                    ;      (80x25 text, mono)
00002173 7405           <1>       je     short _n0 ; same with mode 3 for TRDOS 386
00002175 E9F9070000     <1>       jmp    GRAPHICS_DOWN
                        <1> _n0:
                        <1>       ; 07/07/2016
0000217A 80FF07         <1>       cmp    bh, 7 ; video page number
0000217D 7606           <1>       jna    short n12
0000217F 8A3D[BE810100] <1>       mov    bh, [ACTIVE_PAGE]
                        <1>       ;
                        <1> n12:              ; CONTINUE_DOWN
00002185 88DC           <1>       mov    ah, bl
00002187 6650           <1>       push   ax     ; * ; save attribute in ah
00002189 6689D0         <1>       mov    ax, dx ; LOWER RIGHT CORNER
0000218C E860FFFFFF     <1>       call   scroll_position     ; GET REGEN LOCATION
00002191 741F           <1>       jz     short n14
00002193 29CE           <1>       sub    esi, ecx  ; SI IS FROM ADDRESS
00002195 88F5           <1>       mov    ch, dh  ; #rows in block
00002197 28C5           <1>       sub    ch, al ; #rows to be moved
                        <1> n13:
00002199 E897FFFFFF     <1>       call   n10    ; MOVE ONE ROW
                        <1>
0000219E 51             <1>       push   ecx
0000219F 8A0D[BC6F0000] <1>       mov    cl, [CRT_COLS]
000021A5 00C9           <1>       add    cl, cl
000021A7 29CE           <1>         sub  esi, ecx  ; next line
000021A9 29CF           <1>         sub  edi, ecx
000021AB 59             <1>         pop  ecx
                        <1>
000021AC FECD           <1>       dec    ch      ; count of lines to move
000021AE 75E9           <1>       jnz    short n13 ; row loop
                        <1>       ; ch = 0
000021B0 88C6           <1>       mov    dh, al  ; #rows
                        <1> n14:
                        <1>       ; attribute in ah
000021B2 B020           <1>       mov    al, ' '       ; fill with blanks
                        <1> n15:
000021B4 E889FFFFFF     <1>       call   n11 ; 16/01/2016
                        <1>
000021B9 8A0D[BC6F0000] <1>       mov    cl, [CRT_COLS]
000021BF 00C9           <1>       add    cl, cl
000021C1 29CF           <1>         sub  edi, ecx
                        <1>
000021C3 FECE           <1>       dec    dh
000021C5 75ED           <1>       jnz    short n15
                        <1>       ;
                        <1>       ; 18/11/2020
000021C7 FC             <1>       cld    ; clear direction flag
                        <1>       ;
000021C8 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
000021CD 803D[BA6F0000]07       <1>         cmp     byte [CRT_MODE], 7 ; 6!?
000021D4 7607                   <1>       jna     short read_ac_c
000021D6 31C0                   <1>       xor     eax, eax
000021D8 E97FF9FFFF             <1>         jmp     _video_return
                                <1> read_ac_c:
000021DD E805000000             <1>       call    _read_ac_current
                                <1>       ; 12/05/2016
                                <1>         ;jmp     VIDEO_RETURN
000021E2 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>
000021E7 8A25[BA6F0000]         <1>       mov     ah, [CRT_MODE] ; current video mode
000021ED 80FC04                 <1>       cmp     ah, 4
000021F0 720A                   <1>       jb      short p10
                                <1>       ; 18/11/2020
                                <1>       ;cmp   byte [CRT_MODE], 4
                                <1>       ;jnb   GRAPHICS_READ ; 26/06/2016
                                <1>
000021F2 80FC07                 <1>       cmp     ah, 7 ; TEST FOR BW CARD (80x25 monochrome text)
000021F5 7405                   <1>       je      short p10      ; same with mode 3 in TRDOS 386
000021F7 E9CD080000             <1>       jmp     GRAPHICS_READ
                                <1> p10:
000021FC 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
00002201 803D[BA6F0000]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)
00002208 7516                   <1>       jne     short p13
                                <1>
                                <1>       ; 05/12/2020
0000220A 3A3D[BE810100]         <1>       cmp     bh, [ACTIVE_PAGE]
00002210 750E                   <1>       jne     short p13
                                <1>
                                <1>       ; WAIT FOR HORIZONTAL RETRACE OR VERTICAL RETRACE IF COLOR 80
                                <1> p11:
00002212 FB                     <1>       sti           ; enable interrupts first
                                <1>       ; 05/12/2020
00002213 90                     <1>       nop
                                <1>       ; 18/11/2020
                                <1>       ;or    bl, bl
                                <1>       ;jnz   short p13 ; mode 01h (and mode 00h) - 40x25 color text
00002214 FA                     <1>       cli           ; block interrupts for single loop
00002215 EC                     <1>       in      al, dx ; get status from the adapter
00002216 A801                   <1>       test    al, RHRZ ; is horizontal retrace low
00002218 75F8                   <1>       jnz     short p11 ; wait until it is
                                <1> p12:                ;  wait for either retrace high
0000221A EC                     <1>       in      al, dx ; get status again
0000221B A809                   <1>       test    al, RVRT+RHRZ ; is horizontal or vertical retrace high
0000221D 74FB                   <1>       jz      short p12 ; wait until either retrace active
                                <1> ;p14:
0000221F FB                     <1>       sti
                                <1> p13:
00002220 81C600800B00           <1>       add     esi, 0B8000h
00002226 668B06                 <1>       mov     ax, [esi]
                                <1>
00002229 C3                     <1>       retn   ; 18/01/2016
                                <1>
                                <1> WRITE_AC_CURRENT:
                                <1>       ; 08/07/2016
                                <1>       ; 26/06/2016
                                <1>       ; 24/06/2016
                                <1>       ; 12/05/2016
                                <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
```

```
                                <1>       ;
                                <1>       ; VIDEO.ASM - 06/10/85  VIDEO DISPLAY BIOS
                                <1>       ;
                                <1> ;--------------------------------------------------------------------
                                <1> ; WRITE_AC_CURRENT                                  :
                                <1> ;     THTS ROUTINE WRITES THE ATTRIBUTE AND CHARACTER            :
                                <1> ;     AT THE CURRENT CURSOR POSITION                      :
                                <1> ; INPUT                                                   :
                                <1> ;     (AH) = CURRENT CRT MODE                             :
                                <1> ;     (BH) = DISPLAY PAGE                           :
                                <1> ;     (CX) = COUNT OF CHARACTERS TO WRITE                 :
                                <1> ;     (AL) = CHAR TO WRITE                                :
                                <1> ;     (BL) = ATTRIBUTE OF CHAR TO WRITE                   :
                                <1> ;     (DS) = DATA SEGMENT                           :
                                <1> ;     (ES) = REGEN SEGMENT                                :
                                <1> ; OUTPUT                                            :
                                <1> ;     DISPLAY REGEN BUFFER UPDATED                        :
                                <1> ;--------------------------------------------------------------------
                                <1>
                                <1>       ; 08/07/2016
0000222A 803D[BA6F0000]07       <1>       cmp   byte [CRT_MODE], 7 ; 6!?
00002231 760A                   <1>       jna   short write_ac_c
                                <1>
00002233 E8070B0000             <1>       call  vga_write_char_attr
00002238 E91AF9FFFF             <1>       jmp     VIDEO_RETURN
                                <1>
                                <1> write_ac_c:
0000223D E834000000             <1>       call  _write_c_current
                                <1>
00002242 0FB6F7                 <1>       movzx esi, bh ; video page number (0 to 7)
00002245 889E[C36F0000]         <1>       mov   [esi+chr_attrib], bl ; color/attribute
                                <1>
0000224B E907F9FFFF             <1>         jmp     VIDEO_RETURN
                                <1>
                                <1> WRITE_C_CURRENT:
                                <1>       ; 08/07/2016
                                <1>       ; 26/06/2016
                                <1>       ; 12/05/2016
                                <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>       ;
                                <1>       ; VIDEO.ASM - 06/10/85  VIDEO DISPLAY BIOS
                                <1>       ;
                                <1> ;--------------------------------------------------------------------
                                <1> ; WRITE_C_CURRENT                                   :
                                <1> ;     THIS ROUTINE WRITES THE CHARACTER AT                :
                                <1> ;     THE CURRENT CURSOR POSITION, ATTRIBUTE UNCHANGED    :
                                <1> ; INPUT                                                   :
                                <1> ;     (AH) = CURRENT CRT MODE                             :
                                <1> ;     (BH) = DISPLAY PAGE                           :
                                <1> ;     (CX) = COUNT OF CHARACTERS TO WRITE                 :
                                <1> ;     (AL) = CHAR TO WRITE                                :
                                <1> ;     (DS) = DATA SEGMENT                           :
                                <1> ;     (ES) = REGEN SEGMENT                                :
                                <1> ; OUTPUT                                            :
                                <1> ;     DISPLAY REGEN BUFFER UPDATED                        :
                                <1> ;--------------------------------------------------------------------
                                <1>
                                <1>       ; 08/07/2016
00002250 803D[BA6F0000]07       <1>       cmp   byte [CRT_MODE], 7 ; 6!?
00002257 760A                   <1>       jna   short write_c_c
                                <1>
00002259 E8E10A0000             <1>       call  vga_write_char_only
0000225E E9F4F8FFFF             <1>       jmp     VIDEO_RETURN
                                <1>
                                <1> write_c_c:
                                <1>       ;and  bh, 7 ; video page number (<= 7)
00002263 0FB6F7                 <1>       movzx esi, bh
00002266 8A9E[C36F0000]         <1>       mov   bl, [esi+chr_attrib]
                                <1>
0000226C E805000000             <1>       call  _write_c_current
00002271 E9E1F8FFFF             <1>         jmp     VIDEO_RETURN
                                <1>
                                <1> _write_c_current:  ; from 'write_tty'
                                <1>       ; 18/11/2020
                                <1>       ; 26/06/2016
                                <1>       ; 24/06/2016
                                <1>       ; 12/05/2016
                                <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>       ; 30/08/2014 (Retro UNIX 386 v1)
                                <1>       ; 18/01/2014
                                <1>       ; 04/12/2013
                                <1>       ;
                                <1>       ; VIDEO.ASM - 06/10/85 VIDEO DISPLAY BIOS
                                <1>
                                <1>       ; 18/11/2020
00002276 8A25[BA6F0000]         <1>       mov   ah, [CRT_MODE] ; current video mode
0000227C 80FC04                 <1>       cmp   ah, 4
0000227F 720A                   <1>       jb    short p40
                                <1>
                                <1>       ;cmp  byte [CRT_MODE], 4
                                <1>         ;jnb GRAPHICS_WRITE ; 26/06/2016
                                <1>
00002281 80FC07                 <1>       cmp   ah, 7 ; TEST FOR BW CARD
00002284 7405                   <1>       je    short p40
00002286 E98E070000             <1>       jmp   GRAPHICS_WRITE
                                <1> p40:
                                <1>       ; al = character
                                <1>       ; bl = color/attribute
                                <1>       ; bh = video page
                                <1>       ; cx = count of characters to write
0000228B 6652                   <1>       push  dx
0000228D 88DC                   <1>       mov   ah, bl  ; color/attribute (12/05/2016)
0000228F 6650                   <1>       push  ax    ; save character & attribute/color
00002291 6651                   <1>       push  cx
00002293 E85DFDFFFF             <1>       call  find_position  ; get regen location and port address
```

```
3107 00002298 6659                <1>       pop    cx
3108                              <1>       ; esi = regen location
3109                              <1>       ; dx = status port
3110                              <1>       ;
3111 0000229A 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 000022A0 803D[BA6F0000]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 000022A7 751A                <1>       jne    short p44
3130                              <1> p46:
3131                              <1>       ; 05/12/2020
3132 000022A9 3A3D[BE810100]      <1>       cmp     bh, [ACTIVE_PAGE]
3133 000022AF 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 000022B1 FB                  <1>       sti          ; enable interrupts first
3139 000022B2 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 000022B3 FA                  <1>       cli          ; block interrupts for single loop
3144 000022B4 EC                  <1>       in     al, dx ; get status from the adapter
3145 000022B5 A808                <1>       test   al, RVRT ; check for vertical retrace first
3146 000022B7 7509                <1>       jnz    short p43 ; Do fast write now if vertical retrace
3147 000022B9 A801                <1>       test   al, RHRZ ; is horizontal retrace low
3148 000022BB 75F4                <1>       jnz    short p41 ; wait until it is
3149                              <1> p42:              ;  wait for either retrace high
3150 000022BD EC                  <1>       in     al, dx ; get status again
3151 000022BE A809                <1>       test   al, RVRT+RHRZ ; is horizontal or vertical retrace high
3152 000022C0 74FB                <1>       jz     short p42 ; wait until either retrace active
3153                              <1> p43:
3154 000022C2 FB                  <1>       sti
3155                              <1> p44:
3156 000022C3 668B0424            <1>       mov    ax, [esp] ; restore the character (al) & attribute (ah)
3157 000022C7 668906              <1>       mov    [esi], ax
3158                              <1>
3159 000022CA 6649                <1>       dec    cx
3160 000022CC 740D                <1>       jz     short p45
3161                              <1>
3162 000022CE 46                  <1>       inc    esi
3163 000022CF 46                  <1>       inc    esi
3164                              <1>
3165                              <1>       ; 05/12/2020
3166 000022D0 803D[BA6F0000]03    <1>       cmp    byte [CRT_MODE], 03h
3167 000022D7 75EA                <1>       jne    short p44
3168                              <1>       ;jmp   short p41
3169 000022D9 EBCE                <1>       jmp    short p46
3170                              <1> p45:
3171 000022DB 6658                <1>       pop    ax
3172 000022DD 665A                <1>       pop    dx
3173 000022DF 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)
3204                              <1>       ; 03/12/2013 (Retro UNIX 8086 v1 - beginning)
3205                              <1>       ; (Modified registers: EAX, EBX, ECX, EDX, ESI, EDI)
3206                              <1>       ;
3207                              <1>       ; INPUT -> AL = Character to be written
3208                              <1>       ;         BL = Color (Forecolor, Backcolor)
3209                              <1>       ;         BH = Video Page (0 to 7)
3210                              <1>
3211                              <1>       ; 09/07/2016
```

```
3212 000022E0 803D[BA6F0000]07    <1>         cmp     byte [CRT_MODE], 7
3213 000022E7 760A                <1>       jna    short write_tty_cga
3214                              <1>
3215 000022E9 E83C0D0000          <1>       call   vga_write_teletype
3216 000022EE E964F8FFFF          <1>       jmp    VIDEO_RETURN
3217                              <1>
3218                              <1> write_tty_cga:
3219                              <1>       ; 13/05/2016
3220                              <1>       ;call  _write_tty
3221                              <1>       ; 01/07/2016
3222 000022F3 E818000000          <1>       call   _write_tty_m3
3223 000022F8 E95AF8FFFF          <1>       jmp    VIDEO_RETURN
3224                              <1>
3225                              <1> RVRT  equ    00001000b    ; VIDEO VERTICAL RETRACE BIT
3226                              <1> RHRZ  equ    00000001b    ; VIDEO HORIZONTAL RETRACE BIT
3227                              <1>
3228                              <1> ; Derived from "WRITE_TTY" procedure of IBM "pc-at" rombios source code
3229                              <1> ; (06/10/1985), 'video.asm', INT 10H, VIDEO_IO
3230                              <1> ;
3231                              <1> ; 06/10/85  VIDEO DISPLAY BIOS
3232                              <1> ;
3233                              <1> ;--- WRITE_TTY ----------------------------------------------------------------
3234                              <1> ;                                                      :
3235                              <1> ;   THIS INTERFACE PROVIDES A TELETYPE LIKE INTERFACE TO THE                :
3236                              <1> ;   VIDEO CARDS. THE INPUT CHARACTER IS WRITTEN TO THE CURRENT              :
3237                              <1> ;   CURSOR POSITION, AND THE CURSOR IS MOVED TO THE NEXT POSITION.          :
3238                              <1> ;   IF THE CURSOR LEAVES THE LAST COLUMN OF THE FIELD, THE COLUMN           :
3239                              <1> ;   IS SET TO ZERO, AND THE ROW VALUE IS INCREMENTED. IF THE ROW       :
3240                              <1> ;   ROW VALUE LEAVES THE FIELD, THE CURSOR IS PLACED ON THE LAST ROW,       :
3241                              <1> ;   FIRST COLUMN, AND THE ENTIRE SCREEN IS SCROLLED UP ONE LINE.       :
3242                              <1> ;   WHEN THE SCREEN IS SCROLLED UP, THE ATTRIBUTE FOR FILLING THE           :
3243                              <1> ;   NEWLY BLANKED LINE IS READ FROM THE CURSOR POSITION ON THE PREVIOUS     :
3244                              <1> ;   LINE BEFORE THE SCROLL, IN CHARACTER MODE. IN GRAPHICS MODE,       :
3245                              <1> ;   THE 0 COLOR IS USED.                                              :
3246                              <1> ;   ENTRY --                                                          :
3247                              <1> ;     (AH) = CURRENT CRT MODE                                             :
3248                              <1> ;     (AL) = CHARACTER TO BE WRITTEN                                      :
3249                              <1> ;         NOTE THAT BACK SPACE, CARRIAGE RETURN, BELL AND LINE FEED ARE :
3250                              <1> ;         HANDLED AS COMMANDS RATHER THAN AS DISPLAY GRAPHICS CHARACTERS     :
3251                              <1> ;     (BL) = FOREGROUND COLOR FOR CHAR WRITE IF CURRENTLY IN A GRAPHICS MODE  :
3252                              <1> ;   EXIT --                                                           :
3253                              <1> ;     ALL REGISTERS SAVED                                             :
3254                              <1> ;-------------------------------------------------------------------------------
3255                              <1>
3256                              <1> ; 18/11/2020 (TRDOS 386 v2.0.3)
3257                              <1> ; 09/12/2017
3258                              <1> ; 08/07/2016
3259                              <1> ; 26/06/2016
3260                              <1> ; 24/06/2016
3261                              <1> _write_tty:
3262                              <1>       ; 13/05/2016
3263                              <1>       ; --- 18/11/2020 ---
3264                              <1>       ; NOTE:
3265                              <1>       ; Only kernel calls "_write_tty" procedure...
3266                              <1>       ; TRDOS 386 v2 kernel uses video mode 3 for displaying
3267                              <1>       ; (some error) messages and also mainprog command interpreter
3268                              <1>       ; (in kernel) uses "_write_tty".
3269                              <1>       ; So, here video mode must be set to 3 if it is not 3.
3270                              <1>
3271 000022FD FA                 <1>       cli    ; disable interrupts
3272                              <1>       ;
3273                              <1>       ; 01/09/2014
3274 000022FE 803D[BA6F0000]03    <1>       cmp    byte [CRT_MODE], 3
3275 00002305 7409                <1>       je     short _write_tty_m3
3276                              <1>       ;
3277                              <1> set_mode_3:
3278 00002307 53                  <1>       push   ebx
3279 00002308 50                  <1>       push   eax
3280                              <1>       ;call  _set_mode
3281                              <1>       ; 18/11/2020
3282 00002309 E858F8FFFF          <1>       call   set_txt_mode  ; set video mode to 03h
3283 0000230E 58                  <1>       pop    eax
3284 0000230F 5B                  <1>       pop    ebx
3285                              <1>       ;
3286                              <1> _write_tty_m3: ; 24/06/2016 (m3 -> _write_tty_m3)
3287 00002310 0FB6F7              <1>       movzx  esi, bh ; 12/05/2016
3288 00002313 66D1E6              <1>       shl    si, 1
3289 00002316 81C6[AE810100]      <1>       add    esi, CURSOR_POSN
3290 0000231C 668B16              <1>       mov    dx, [esi]
3291                              <1>       ;
3292                              <1>       ; dx now has the current cursor position
3293                              <1>       ;
3294 0000231F 3C0D                <1>       cmp    al, 0Dh      ; CR   ; is it carriage return or control character
3295 00002321 763E                <1>       jbe    short u8
3296                              <1>       ;
3297                              <1>       ; write the char to the screen
3298                              <1> u0:
3299                              <1>       ; al = character
3300                              <1>       ; bl = attribute/color
3301                              <1>       ; bh = video page number (0 to 7)
3302                              <1>       ;
3303 00002323 66B90100            <1>       mov    cx, 1  ; 24/06/2016
3304                              <1>       ; cx = count of characters to write
3305                              <1>       ;
3306 00002327 E84AFFFFFF          <1>       call   _write_c_current ; 16/01/2015
3307                              <1>       ;
3308                              <1>       ; position the cursor for next char
3309 0000232C FEC2                <1>       inc    dl           ; next column
3310 0000232E 3A15[BC6F0000]      <1>       cmp    dl, [CRT_COLS]  ; test for column overflow
3311 00002334 7561                <1>         jne     _set_cpos
3312 00002336 B200                <1>       mov    dl, 0        ; column = 0
3313                              <1> u10:                      ; (line feed found)
3314 00002338 80FE18              <1>       cmp    dh, 25-1     ; check for last row
3315 0000233B 7220                <1>       jb     short u6
3316                              <1>       ;
```

```
                        <1>       ; scroll required
                        <1> u1:
                        <1>       ; SET CURSOR POSITION (04/12/2013)
0000233D E855000000     <1>       call  _set_cpos
                        <1>       ;
                        <1>       ; determine value to fill with during scroll
                        <1> u2:
                        <1>       ; bh = video page number
                        <1>       ;
00002342 E8A0FEFFFF     <1>       call  _read_ac_current ; 18/01/2016
                        <1>       ;
                        <1>       ; al = character, ah = attribute
                        <1>       ; bh = video page number
                        <1>       ; 18/11/2020
00002347 88E3           <1>       mov   bl, ah ; color/attribute
                        <1> u3:
                        <1>       ;;mov  ax, 0601h    ; scroll one line
                        <1>       ;;sub  cx, cx       ; upper left corner
                        <1>       ;;mov  dh, 25-1     ; lower right row
                        <1>       ;;;mov dl, [CRT_COLS]
                        <1>       ;mov   dl, 80       ; lower right column
                        <1>       ;;dec  dl
                        <1>       ;;mov  dl, 79
                        <1>
                        <1>       ;;call scroll_up    ; 04/12/2013
                        <1>       ;;; 11/03/2015
                        <1>       ; 02/09/2014
                        <1>       ;;;mov cx, [crt_ulc] ; Upper left corner  (0000h)
                        <1>       ;;;mov dx, [crt_lrc] ; Lower right corner (184Fh)
                        <1>       ; 11/03/2015
00002349 6629C9         <1>       sub   cx, cx
                        <1>       ;mov   dx, 184Fh ; dl = 79 (column), dh = 24 (row)
                        <1>       ; 18/11/2020
0000234C B618           <1>       mov   dh, 25-1
0000234E 8A15[BC6F0000] <1>       mov   dl, [CRT_COLS]
00002354 FECA           <1>       dec   dl
                        <1>       ;
00002356 B001           <1>       mov   al, 1        ; scroll 1 line up
                        <1>             ; ah = attribute
                        <1>       ;mov  bl, al ; 12/05/2016
00002358 E9E0FCFFFF     <1>       jmp   _scroll_up   ; 16/01/2016
                        <1> ;u4:
                        <1>       ;;int  10h          ; video-call return
                        <1>                           ; scroll up the screen
                        <1>                           ; tty return
                        <1> ;u5:
                        <1>       ;retn               ; return to the caller
                        <1>
                        <1> u6:                       ; set-cursor-inc
0000235D FEC6           <1>       inc   dh            ; next row
                        <1>                           ; set cursor
                        <1> ;u7:
                        <1>       ;;mov  ah, 02h
                        <1>       ;;jmp  short u4     ; establish the new cursor
                        <1>       ;call  _set_cpos
                        <1>       ;jmp   short u5
0000235F EB36           <1>       jmp     _set_cpos
                        <1>
                        <1>       ; check for control characters
                        <1> u8:
00002361 7432           <1>       je     short u9
00002363 3C0A           <1>       cmp    al, 0Ah             ; is it a line feed (0Ah)
00002365 74D1           <1>       je     short u10
00002367 3C07           <1>       cmp    al, 07h      ; is it a bell
00002369 7476           <1>       je     short u11
0000236B 3C08           <1>       cmp    al, 08h             ; is it a backspace
                        <1>       ;jne   short u0
0000236D 741E           <1>       je     short bs     ; 12/12/2013
                        <1>       ; 12/12/2013 (tab stop)
0000236F 3C09           <1>       cmp    al, 09h             ; is it a tab stop
00002371 75B0           <1>       jne    short u0
00002373 88D0           <1>       mov    al, dl
                        <1>       ;cbw
00002375 30E4           <1>       xor    ah, ah ; 09/12/2017
00002377 B108           <1>       mov    cl, 8
00002379 F6F1           <1>       div    cl
0000237B 28E1           <1>       sub    cl, ah
                        <1> ts:
                        <1>       ; 02/09/2014
                        <1>       ; 01/09/2014
                        <1>       ;mov   al, 20h
                        <1> tsloop:
0000237D 6651           <1>       push   cx
                        <1>       ; 18/11/2020
                        <1>       ;push  ax
                        <1>       ;;mov  bh, [ACTIVE_PAGE]
                        <1>       ; 05/12/2020
                        <1>       ;push  bx
0000237F B020           <1>       mov    al, 20h ; al = space (blank)
                        <1>                      ; bl = color/attribute
00002381 E88AFFFFFF     <1>       call   _write_tty_m3 ; 24/06/2016 (m3 -> _write_tty_m3)
                        <1>       ; 05/12/2020
                        <1>       ; bx is preserved in '_write_tty_m3'
                        <1>       ; 18/11/2020
                        <1>       ;pop   bx  ; BL = color/attribute, Bh = video page
                        <1>       ;pop   ax  ; AL = character
00002386 6659           <1>       pop    cx
00002388 FEC9           <1>       dec    cl
0000238A 75F1           <1>       jnz    short tsloop
0000238C C3             <1>       retn
                        <1> bs:
                        <1>       ; back space found
                        <1>
0000238D 08D2           <1>       or     dl, dl              ; is it already at start of line
                        <1>       ;je    short u7     ; set_cursor
```

```
3422 0000238F 7406              <1>       jz     short _set_cpos
3423 00002391 664A              <1>       dec    dx                  ; no -- just move it back
3424                            <1>       ;jmp   short u7
3425 00002393 EB02              <1>       jmp    short _set_cpos
3426                            <1>
3427                            <1>       ; carriage return found
3428                            <1> u9:
3429 00002395 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 00002397 BE[AE810100]      <1>       mov    esi, CURSOR_POSN
3461 0000239C 0FB6C7            <1>         movzx   eax, bh   ; BH = video page number
3462                            <1> ;     or     al, al
3463                            <1> ;     jz     short _set_cpos_0
3464 0000239F D0E0              <1>         shl     al, 1   ; word offset
3465 000023A1 01C6              <1>         add     esi, eax
3466                            <1> ;_set_cpos_0:
3467 000023A3 668916            <1>       mov    [esi], dx ; save the pointer
3468 000023A6 383D[BE810100]    <1>       cmp    [ACTIVE_PAGE], bh
3469 000023AC 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 000023AE E830FCFFFF        <1>       call   position ; determine location in regen buffer
3477 000023B3 668B0D[AC810100]  <1>       mov    cx, [CRT_START]
3478 000023BA 6601C1            <1>       add    cx, ax  ; add char position in regen buffer
3479                            <1>                    ; to the start address (offset) for this page
3480 000023BD 66D1E9            <1>       shr    cx, 1  ; divide by 2 for char only count
3481 000023C0 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 000023C2 FA                <1>       cli
3489                            <1>       ;mov   dx, [addr_6845] ; address register
3490 000023C3 66BAD403          <1>       mov    dx, 03D4h ; I/O address of color card
3491 000023C7 88E0              <1>       mov    al, ah ; get value
3492 000023C9 EE                <1>       out    dx, al ; register set
3493 000023CA 6642              <1>       inc    dx     ; data register
3494 000023CC EB00              <1>       jmp    $+2    ; i/o delay
3495 000023CE 88E8              <1>       mov    al, ch ; data
3496 000023D0 EE                <1>       out    dx, al
3497 000023D1 664A              <1>       dec    dx
3498 000023D3 88E0              <1>       mov    al, ah
3499 000023D5 FEC0              <1>       inc    al     ; point to other data register
3500 000023D7 EE                <1>       out    dx, al ; set for second register
3501 000023D8 6642              <1>       inc    dx
3502 000023DA EB00              <1>       jmp    $+2    ; i/o delay
3503 000023DC 88C8              <1>       mov    al, cl ; second data value
3504 000023DE EE                <1>       out    dx, al
3505 000023DF FB                <1>       sti
3506                            <1> m17:
3507 000023E0 C3                <1>       retn
3508                            <1>
3509                            <1> beeper:
3510                            <1>       ; 04/08/2016
3511                            <1>       ; 12/05/2016 - TRDOS 386 (TRDOS v2.0)
3512                            <1>       ; 30/08/2014 (Retro UNIX 386 v1)
3513                            <1>       ; 18/01/2014
3514                            <1>       ; 03/12/2013
3515                            <1>       ; bell found
3516                            <1> u11:
3517 000023E1 FB                <1>       sti
3518 000023E2 3A3D[BE810100]    <1>       cmp    bh, [ACTIVE_PAGE]
3519 000023E8 7551              <1>       jne    short u12     ; Do not sound the beep
3520                            <1>                            ; if it is not written on the active page
3521                            <1> beeper_gfx: ; 04/08/2016
3522 000023EA 66B93305          <1>       mov    cx, 1331      ; divisor for 896 hz tone
3523 000023EE B31F              <1>       mov    bl, 31        ; set count for 31/64 second for beep
3524                            <1>       ;call  beep          ; sound the pod bell
3525                            <1>       ;jmp   short u5      ; tty_return
3526                            <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>       ; 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>
000023F0 9C                     <1>       pushf  ; 18/01/2014 ; save interrupt status
000023F1 FA                     <1>       cli                 ; block interrupts during update
000023F2 B0B6                   <1>       mov   al, 10110110b; select timer 2, lsb, msb binary
000023F4 E643                   <1>       out   TIMER+3, al   ; write timer mode register
000023F6 EB00                   <1>       jmp   $+2           ; I/O delay
000023F8 88C8                   <1>       mov   al, cl        ; divisor for hz (low)
000023FA E642                   <1>       out   TIMER+2,AL    ; write timer 2 count - lsb
000023FC EB00                   <1>       jmp   $+2           ; I/O delay
000023FE 88E8                   <1>       mov   al, ch        ; divisor for hz (high)
00002400 E642                   <1>       out   TIMER+2, al   ; write timer 2 count - msb
00002402 E461                   <1>       in    al, PORT_B    ; get current setting of port
00002404 88C4                   <1>       mov   ah, al        ; save that setting
00002406 0C03                   <1>       or    al, GATE2+SPK2      ; gate timer 2 and turn speaker on
00002408 E661                   <1>       out   PORT_B, al    ; and restore interrupt status
                                <1>       ;popf  ; 18/01/2014
0000240A FB                     <1>       sti
                                <1> g7:                       ; 1/64 second per count (bl)
0000240B B90B040000             <1>       mov   ecx, 1035     ; delay count for 1/64 of a second
00002410 E827000000             <1>       call  waitf         ; go to beep delay 1/64 count
00002415 FECB                   <1>       dec   bl            ; (bl) length count expired?
00002417 75F2                   <1>       jnz   short g7      ; no - continue beeping speaker
                                <1>       ;
                                <1>       ;pushf              ; save interrupt status
00002419 FA                     <1>       cli   ; 18/01/2014 ; block interrupts during update
0000241A E461                   <1>       in    al, PORT_B    ; get current port value
                                <1>         ;or      al, not (GATE2+SPK2) ; isolate current speaker bits in case
0000241C 0CFC                   <1>         or      al, ~(GATE2+SPK2)
0000241E 20C4                   <1>         and ah, al        ; someone turned them off during beep
00002420 88E0                   <1>       mov   al, ah        ; recover value of port
                                <1>         ;or      al, not (GATE2+SPK2) ; force speaker data off
00002422 0CFC                   <1>       or    al, ~(GATE2+SPK2) ; isolate current speaker bits in case
00002424 E661                   <1>       out   PORT_B, al    ; and stop speaker timer
                                <1>       ;popf               ; restore interrupt flag state
00002426 FB                     <1>       sti
00002427 B90B040000             <1>       mov   ecx, 1035     ; force 1/64 second delay (short)
0000242C E80B000000             <1>       call  waitf         ; minimum delay between all beeps
                                <1>       ;pushf              ; save interrupt status
00002431 FA                     <1>       cli                 ; block interrupts during update
00002432 E461                   <1>       in    al, PORT_B    ; get current port value in case
00002434 2403                   <1>       and   al, GATE2+SPK2      ; someone turned them on
00002436 08E0                   <1>       or    al, ah        ; recover value of port_b
00002438 E661                   <1>       out   PORT_B, al    ; restore speaker status
0000243A 9D                     <1>       popf                ; restore interrupt flag state
                                <1> u12:
0000243B C3                     <1>       retn
                                <1>
                                <1> REFRESH_BIT equ    00010000b    ; REFRESH TEST BIT
                                <1>
                                <1> WAITF:
                                <1> waitf:
                                <1>       ; 30/08/2014 (Retro UNIX 386 v1)
                                <1>       ; 03/12/2013
                                <1>       ;
                                <1> ;     push ax                   ; save work register (ah)
                                <1> ;waitf1:
                                <1>                           ; use timer 1 output bits
                                <1> ;     in    al, PORT_B    ; read current counter output status
                                <1> ;     and   al, REFRESH_BIT     ; mask for refresh determine bit
                                <1> ;     cmp   al, ah        ; did it just change
                                <1> ;     je    short waitf1 ; wait for a change in output line
                                <1> ;     ;
                                <1> ;     mov   ah, al        ; save new lflag state
                                <1> ;     loop  waitf1        ; decrement half cycles till count end
                                <1> ;     ;
                                <1> ;     pop   ax            ; restore (ah)
                                <1> ;     retn                ; return (cx)=0
                                <1>
                                <1> ; 06/02/2015 (unix386.s <-- dsectrm2.s)
                                <1> ; 17/12/2014 (dsectrm2.s)
                                <1> ; WAITF
                                <1> ; /// IBM PC-XT Model 286 System BIOS Source Code - Test 4 - 06/10/85 ///
                                <1> ;
                                <1> ;---WAITF--------------------------------------------------------------------
                                <1> ;     FIXED TIME WAIT ROUTINE (HARDWARE CONTROLLED - NOT PROCESSOR)
                                <1> ; ENTRY:
                                <1> ;     (CX) = COUNT OF 15.085737 MICROSECOND INTERVALS TO WAIT
                                <1> ;            MEMORY REFRESH TIMER 1 OUTPUT USED AS REFERENCE
                                <1> ; EXIT:
                                <1> ;                  AFTER (CX) TIME COUNT (PLUS OR MINUS 16 MICROSECONDS)
                                <1> ;     (CX) = 0
                                <1> ;--------------------------------------------------------------------------
                                <1>
                                <1> ; Refresh period: 30 micro seconds (15-80 us)
```

```
3632                                    <1> ; (16/12/2014 - AWARDBIOS 1999 - ATORGS.ASM, WAIT_REFRESH)
3633                                    <1>
3634                                    <1> ;WAITF:                                 ; DELAY FOR (CX)*15.085737 US
3635 0000243C 6650                     <1>         PUSH   AX                   ; SAVE WORK REGISTER (AH)
3636                                    <1>         ; 16/12/2014
3637                                    <1>         ;shr   cx, 1                ; convert to count of 30 micro seconds
3638 0000243E D1E9                     <1>         shr    ecx, 1 ; 21/02/2015
3639                                    <1> ;17/12/2014
3640                                    <1> ;WAITF1:
3641                                    <1> ;       IN     AL, PORT_B   ;061h   ; READ CURRENT COUNTER OUTPUT STATUS
3642                                    <1> ;       AND    AL, REFRESH_BIT      ;00010000b ; MASK FOR REFRESH DETERMINE BIT
3643                                    <1> ;       CMP    AL, AH               ; DID IT JUST CHANGE
3644                                    <1> ;       JE     short WAITF1         ; WAIT FOR A CHANGE IN OUTPUT LINE
3645                                    <1> ;       MOV    AH, AL               ; SAVE NEW FLAG STATE
3646                                    <1> ;       LOOP   WAITF1               ; DECREMENT HALF CYCLES TILL COUNT END
3647                                    <1>         ;
3648                                    <1>         ; 17/12/2014
3649                                    <1>         ;
3650                                    <1>         ; Modification from 'WAIT_REFRESH' procedure of AWARD BIOS - 1999
3651                                    <1>         ;
3652                                    <1> ;WAIT_REFRESH:  Uses port 61, bit 4 to have CPU speed independent waiting.
3653                                    <1> ;       INPUT:  CX = number of refresh periods to wait
3654                                    <1> ;                   (refresh periods = 1 per 30 microseconds on most machines)
3655                                    <1> WR_STATE_0:
3656 00002440 E461                     <1>         IN     AL,PORT_B            ; IN AL,SYS1
3657 00002442 A810                     <1>         TEST   AL,010H
3658 00002444 74FA                     <1>         JZ     SHORT WR_STATE_0
3659                                    <1> WR_STATE_1:
3660 00002446 E461                     <1>         IN     AL,PORT_B            ; IN AL,SYS1
3661 00002448 A810                     <1>         TEST   AL,010H
3662 0000244A 75FA                     <1>         JNZ    SHORT WR_STATE_1
3663 0000244C E2F2                     <1>           LOOP    WR_STATE_0
3664                                    <1>         ;
3665 0000244E 6658                     <1>         POP    AX                   ; RESTORE (AH)
3666 00002450 C3                       <1>         RETn                        ; (CX) = 0
3667                                    <1>
3668                                    <1> ; 09/07/2016
3669                                    <1> ; 01/07/2016
3670                                    <1> ; 24/06/2016
3671                                    <1> ; 23/06/2016 - TRDOS 386 (TRDOS v2.0)
3672                                    <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
3673                                    <1> ;--------------------------------------------------------------------------------
3674                                    <1> ; WRITE_STRING                                                             :
3675                                    <1> ;       THIS ROUTINE WRITES A STRING OF CHARACTERS TO THE CRT.                  :
3676                                    <1> ; INPUT                                                                    :
3677                                    <1> ;       (AL) = WRITE STRING COMMAND  0 - 3                                  :
3678                                    <1> ;       (BH) = DISPLAY PAGE (ACTIVE PAGE)                                   :
3679                                    <1> ;       (CX) = COUNT OF CHARACTERS TO WRITE, IF (CX) = 0 THEN RETURN            :
3680                                    <1> ;       (DX) = CURSOR POSITION FOR START OF STRING WRITE                     :
3681                                    <1> ;       (BL) = ATTRIBUTE OF CHARACTER TO WRITE IF (AL) = 0  OR     (AL) = 1       :
3682                                    <1> ;       (eBP) = SOURCE STRING OFFSET                                        :
3683                                    <1> ; OUTPUT                                                                   :
3684                                    <1> ;       NONE                                                                :
3685                                    <1> ;--------------------------------------------------------------------------------
3686                                    <1>
3687                                    <1> ; AL = 00h: Assign all characters the attribute in BL; do not update cursor
3688                                    <1> ; AL = 01h: Assign all characters the attribute in BL; update cursor
3689                                    <1> ; AL = 02h: Use attributes in string; do not update cursor
3690                                    <1> ; AL = 03h: Use attributes in string; update cursor
3691                                    <1>
3692                                    <1> WRITE_STRING:
3693                                    <1>         ; 12/09/2016
3694                                    <1>         ; 09/07/2016
3695                                    <1>         ;cmp   byte [CRT_MODE], 7 ; 6?!
3696                                    <1>         ;ja    VIDEO_RETURN         ; not a valid function for VGA modes
3697                                    <1>         ;
3698 00002451 A2[248E0100]             <1>         mov    [w_str_cmd], al           ; save (AL) command
3699 00002456 3C04                     <1>         CMP    AL, 4                ; TEST FOR INVALID WRITE STRING OPTION
3700 00002458 0F83F9F6FFFF             <1>         JNB    VIDEO_RETURN         ; IF OPTION INVALID THEN RETURN
3701                                    <1>
3702                                    <1>           ;JCXZ   VIDEO_RETURN            ; IF ZERO LENGTH STRING THEN RETURN
3703                                    <1>
3704 0000245E 67E362                   <1>           jcxz    P55                   ; 01/07/2016
3705                                    <1>
3706                                    <1>         ; 01/07/2016
3707                                    <1>         ;and   ecx, 0FFFFh
3708                                    <1>         ; ECX = byte count
3709                                    <1>         ;push  ecx
3710 00002461 89EE                     <1>         mov    esi, ebp ; user buffer
3711 00002463 BF00000700               <1>         mov    edi, Cluster_Buffer  ; system buffer
3712 00002468 E8A1EE0000               <1>         call   transfer_from_user_buffer
3713                                    <1>         ;pop   ecx
3714 0000246D 0F82E4F6FFFF             <1>         jc     VIDEO_RETURN
3715                                    <1>         ; ecx = transfer (byte) count = character count
3716 00002473 BD00000700               <1>         mov    ebp, Cluster_Buffer
3717                                    <1>         ; 12/09/2016
3718 00002478 803D[BA6F0000]07         <1>         cmp    byte [CRT_MODE], 7 ; 6?!
3719 0000247F 0F87A1000000             <1>         ja     vga_write_string
3720                                    <1>         ;
3721 00002485 0FB6F7                   <1>         movzx  esi, bh                   ; GET CURRENT CURSOR PAGE
3722 00002488 66D1E6                   <1>         SAL    SI, 1                ; CONVERT TO PAGE OFFSET  (SI= PAGE)
3723                                    <1>         ; *****
3724 0000248B 66FFB6[AE810100]         <1>         PUSH   word [eSI+CURSOR_POSN]    ; SAVE CURRENT CURSOR POSITION IN STACK
3725                                    <1>
3726                                    <1>         ;MOV   AX,0200H             ; SET NEW CURSOR POSITION
3727                                    <1>         ;INT   10H
3728                                    <1> P50next:
3729 00002492 51                       <1>         push   ecx ; ****
3730 00002493 53                       <1>         push   ebx ; *** ; 18/11/2020
3731 00002494 56                       <1>         push   esi ; **
3732 00002495 52                       <1>         push   edx ; *
3733 00002496 E8FCFEFFFF               <1>         call   _set_cpos
3734                                    <1> P50:
3735 0000249B 8A4500                   <1>         MOV    AL, [eBP]             ; GET CHARACTER FROM INPUT STRING
3736 0000249E 45                       <1>         INC    eBP                   ; BUMP POINTER TO CHARACTER
```

```
                                <1>
                                <1> ;-----      TEST FOR SPECIAL CHARACTER'S
                                <1>
0000249F 3C08                   <1>       CMP   AL, 08H                   ; IS IT A BACKSPACE
000024A1 7410                   <1>       JE    short P51           ; BACK_SPACE
000024A3 3C0D                   <1>       CMP   AL, 0Dh ; CR        ; IS IT CARRIAGE RETURN
000024A5 740C                   <1>       JE    short P51           ; CAR_RET
000024A7 3C0A                   <1>       CMP   AL, 0Ah ; LF        ; IS IT A LINE FEED
000024A9 7408                   <1>       JE    short P51           ; LINE_FEED
                                <1>       ; 18/11/2020
000024AB 3C09                   <1>       cmp   al, 09h                   ; is it a tab stop
000024AD 7404                   <1>       je    short P51
                                <1>       ;
000024AF 3C07                   <1>       CMP   AL, 07h                   ; IS IT A BELL
000024B1 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>
000024B3 E858FEFFFF             <1>       call  _write_tty_m3
                                <1>
000024B8 5A                     <1>       pop   edx ; *
000024B9 5E                     <1>       pop   esi ; **
                                <1>
000024BA 668B96[AE810100]       <1>       MOV   DX, [eSI+CURSOR_POSN]     ; GET CURRENT CURSOR POSITION
000024C1 EB44                   <1>       JMP   SHORT P54           ; SET CURSOR POSITION AND CONTINUE
                                <1> P55:
000024C3 E98FF6FFFF             <1>       JMP   VIDEO_RETURN
                                <1> P52:
000024C8 66B90100               <1>       MOV   CX, 1               ; SET CHARACTER WRITE AMOUNT TO ONE
000024CC 803D[248E0100]02       <1>       CMP   byte [w_str_cmd], 2 ; IS THE ATTRIBUTE IN THE STRING
000024D3 7204                   <1>       JB    short P53           ; IF NOT THEN SKIP
000024D5 8A5D00                 <1>       MOV   BL, [eBP]           ; ELSE GET NEW ATTRIBUTE
000024D8 45                     <1>       INC   eBP                 ; BUMP STRING POINTER
                                <1> P53:
                                <1>       ;MOV  AH,09H              ; GOT_CHARACTER
                                <1>       ;INT  10H                 ; WRITE CHARACTER TO THE CRT
                                <1>
000024D9 E898FDFFFF             <1>       call  _write_c_current
                                <1>
000024DE 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>
000024DF 0FB6F7                 <1>       movzx  esi, bh ; video page number (0 to 7)
000024E2 889E[C36F0000]         <1>       mov   [esi+chr_attrib], bl ; color/attribute
                                <1>
000024E8 FEC2                   <1>       INC   DL                  ; INCREMENT COLUMN COUNTER
000024EA 3A15[BC6F0000]         <1>       CMP   DL, [CRT_COLS]            ; IF COLS ARE WITHIN RANGE FOR THIS MODE
                                <1>       ;JB   short P54           ;    THEN GO TO COLUMNS SET
000024F0 7214                   <1>       jb    short P56 ; 05/12/2020
000024F2 FEC6                   <1>       INC   DH                  ; BUMP ROW COUNTER BY ONE
000024F4 28D2                   <1>       SUB   DL, DL              ; SET COLUMN COUNTER TO ZERO
000024F6 80FE19                 <1>       CMP   DH, 25              ; IF ROWS ARE LESS THAN 25 THEN
                                <1>       ;JB   short P54           ; GO TO ROWS_COLUMNS_SET
000024F9 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
000024FB B00A                   <1>       mov   al, 0Ah      ; line feed
                                <1>
000024FD E80EFEFFFF             <1>       call  _write_tty_m3
                                <1>
00002502 66BA0018               <1>       mov   dx, 1800h           ; Column = 0, Row = 24
                                <1> P56:
                                <1>       ; 05/12/2020
                                <1>       ; 18/11/2020
00002506 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
00002507 5B                     <1>       pop   ebx ; ***
00002508 59                     <1>       pop   ecx ; ****
                                <1>
                                <1>       ;LOOP P50                 ; DO IT ONCE MORE UNTIL (CX) = ZERO
00002509 6649                   <1>       dec   cx
0000250B 7585                   <1>       jnz   short P50next
                                <1>
0000250D 665A                   <1>       POP   DX  ; *****         ; RESTORE OLD CURSOR COORDINATES
                                <1>
0000250F F605[248E0100]01       <1>       test  byte [w_str_cmd], 1 ; IF CURSOR WAS NOT TO BE MOVED
00002516 0F853BF6FFFF           <1>       JNZ   VIDEO_RETURN        ; THEN EXIT WITHOUT RESETTING OLD VALUE
                                <1>
                                <1>       ;MOV  AX,0200H            ; ELSE RESTORE OLD CURSOR POSITION
                                <1>       ;INT  10H
                                <1>                                 ; DONE - EXIT WRITE STRING
0000251C E876FEFFFF             <1>       call  _set_cpos
00002521 E931F6FFFF             <1>       JMP   VIDEO_RETURN        ; RETURN TO CALLER
                                <1>
                                <1> vga_write_string:
                                <1>       ; 12/09/2016 - 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_string'
                                <1>
```

```
3842                                    <1>       ; INPUT                                                 :
3843                                    <1>       ;      (AL) = WRITE STRING COMMAND  0 - 3                :
3844                                    <1>       ;      (BH) = DISPLAY PAGE (ACTIVE PAGE)                 :
3845                                    <1>       ;      (CX) = COUNT OF CHARACTERS TO WRITE, IF (CX) = 0 THEN RETURN       :
3846                                    <1>       ;      (DX) = CURSOR POSITION FOR START OF STRING WRITE   :
3847                                    <1>       ;      (BL) = ATTRIBUTE OF CHARACTER TO WRITE IF (AL) = 0  OR     (AL) = 1  :
3848                                    <1>       ;      (eBP) = SOURCE STRING OFFSET                      :
3849                                    <1>       ; OUTPUT                                                :
3850                                    <1>       ;      NONE                                              :
3851                                    <1>       ;-------------------------------------------------------------------------;
3852                                    <1>
3853                                    <1>       ; AL = 00h: Assign all characters the attribute in BL; do not update cursor
3854                                    <1>       ; AL = 01h: Assign all characters the attribute in BL; update cursor
3855                                    <1>       ; AL = 02h: Use attributes in string; do not update cursor
3856                                    <1>       ; AL = 03h: Use attributes in string; update cursor
3857                                    <1>
3858                                    <1>       ; biosfn_write_string(GET_AL(),GET_BH(),GET_BL(),CX,GET_DH(),GET_DL(),ES,BP);
3859                                    <1>       ; static void biosfn_write_string (flag,page,attr,count,row,col,seg,offset)
3860                                    <1>
3861                                    <1>       ; // Read curs info for the page
3862                                    <1>       ; biosfn_get_cursor_pos(page,&dummy,&oldcurs);
3863                                    <1>       ; bh = video page = 0
3864                                    <1>       ;movzx esi, word [CURSOR_POSN] ; current cursor position for video page 0
3865                                    <1>
3866                                    <1>       ; // if row=0xff special case : use current cursor position
3867                                    <1>       ; if(row==0xff)
3868                                    <1>       ;  {col=oldcurs&0x00ff;
3869                                    <1>       ;   row=(oldcurs&0xff00)>>8;
3870                                    <1>       ;  }
3871                                    <1>
3872                                    <1>       ;mov   al, [w_str_cmd]
3873                                    <1>
3874 00002526 80FEFF                   <1>       cmp    dh, 0FFh
3875 00002529 7407                     <1>       je     short vga_wstr_1 ; user current cursor position
3876                                    <1> vga_wstr_0:
3877                                    <1>       ; set cursor position
3878 0000252B 668915[AE810100]         <1>       mov    [CURSOR_POSN], dx ; save cursor pos for pg 0
3879                                    <1> vga_wstr_1:
3880 00002532 66FF35[AE810100]         <1>       push   word [CURSOR_POSN] ; *
3881                                    <1>
3882                                    <1>       ; ebp = string offset in system buffer (user buffer was copied to)
3883                                    <1>
3884                                    <1>       ; while(count--!=0)
3885                                    <1>       ;  {
3886                                    <1>       ;   car=read_byte(seg,offset++);
3887                                    <1>       ;   if((flag&0x02)!=0)
3888                                    <1>       ;    attr=read_byte(seg,offset++);
3889                                    <1>       ;    biosfn_write_teletype(car,page,attr,WITH_ATTR);
3890                                    <1>       ;  }
3891                                    <1>
3892                                    <1>       ;push  eax ; **
3893                                    <1>       ;test  al, 2
3894 00002539 F605[248E0100]02         <1>       test   byte [w_str_cmd], 2
3895 00002540 751D                     <1>       jnz    short vga_wstr_3
3896 00002542 881D[BF810100]           <1>       mov    [ccolor], bl
3897                                    <1> vga_wstr_2:
3898 00002548 51                       <1>       push   ecx
3899 00002549 8A4500                   <1>       mov    al, [ebp]
3900 0000254C E8D90A0000               <1>       call   vga_write_teletype
3901 00002551 59                       <1>       pop    ecx
3902 00002552 6649                     <1>       dec    cx
3903 00002554 741E                     <1>       jz     short vga_wstr_4
3904 00002556 45                       <1>       inc    ebp
3905 00002557 8A1D[BF810100]           <1>       mov    bl, [ccolor]
3906 0000255D EBE9                     <1>       jmp    short vga_wstr_2
3907                                    <1> vga_wstr_3:
3908 0000255F 51                       <1>       push   ecx
3909 00002560 8A4500                   <1>       mov    al, [ebp]
3910 00002563 45                       <1>       inc    ebp
3911 00002564 8A5D00                   <1>       mov    bl, [ebp]
3912 00002567 E8BE0A0000               <1>       call   vga_write_teletype
3913 0000256C 59                       <1>       pop    ecx
3914 0000256D 6649                     <1>       dec    cx
3915 0000256F 7403                     <1>       jz     short vga_wstr_4
3916 00002571 45                       <1>       inc    ebp
3917 00002572 EBEB                     <1>       jmp    short vga_wstr_3
3918                                    <1> vga_wstr_4:
3919                                    <1>       ; // Set back curs pos
3920                                    <1>       ; if((flag&0x01)==0)
3921                                    <1>       ;  biosfn_set_cursor_pos(page,oldcurs);
3922                                    <1>       ; }
3923                                    <1>       ;pop   eax ; **
3924 00002574 665A                     <1>       pop    dx ; word [CURSOR_POSN] ; *
3925                                    <1>       ;test  al, 1
3926 00002576 F605[248E0100]01         <1>       test   byte [w_str_cmd], 1
3927 0000257D 0F85D4F5FFFF             <1>       jnz    VIDEO_RETURN
3928 00002583 668915[AE810100]         <1>       mov    [CURSOR_POSN], dx
3929 0000258A E9C8F5FFFF               <1>       JMP    VIDEO_RETURN
3930                                    <1>
3931                                    <1> ; 07/07/2016
3932                                    <1> ; 27/06/2016 - TRDOS 386 (TRDOS v2.0)
3933                                    <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
3934                                    <1> ;-----------------------------------------------------
3935                                    <1> ;  SCROLL UP
3936                                    <1> ;   THIS ROUTINE SCROLLS UP THE INFORMATION ON THE CRT
3937                                    <1> ; ENTRY ---
3938                                    <1> ;  CH,CL = UPPER LEFT CORNER OF REGION TO SCROLL
3939                                    <1> ;  DH,DL = LOWER RIGHT CORNER OF REGION TO SCROLL
3940                                    <1> ;   BOTH OF THE ABOVE ARE IN CHARACTER POSITIONS
3941                                    <1> ;  BH = FILL VALUE FOR BLANKED LINES
3942                                    <1> ;  AL = # LINES TO SCROLL (AL=0 MEANS BLANK THE ENTIRE FIELD)
3943                                    <1> ;  DS = DATA SEGMENT
3944                                    <1> ;  ES = REGEN SEGMENT
3945                                    <1> ; EXIT --
3946                                    <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]
0000258F 80FC07         <1>       cmp    ah, 7
00002592 7766           <1>       ja     short vga_graphics_up
                        <1>       ;je    n0
                        <1>
00002594 88C7           <1>       MOV    bh, al              ; save line count in BH
00002596 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>
00002599 E8DA050000     <1>       CALL   GRAPH_POSN
0000259E 0FB7F8         <1>       MOVzx  eDI, AX                  ; SAVE RESULT AS DESTINATION ADDRESS
                        <1>
                        <1> ;-----      DETERMINE SIZE OF WINDOW
                        <1>
000025A1 6629CA         <1>       SUB    DX, CX
000025A4 6681C20101     <1>         ADD     DX, 101h               ; ADJUST VALUES
000025A9 C0E602         <1>       SAL    DH, 2               ; MULTIPLY ROWS BY 4 AT 8 VERT DOTS/CHAR
                        <1>                                  ; AND EVEN/ODD ROWS
                        <1> ;-----      DETERMINE CRT MODE
                        <1>
000025AC 803D[BA6F0000]06 <1>       CMP    byte [CRT_MODE], 6  ; TEST FOR MEDIUM RES
000025B3 7305           <1>         JNC     short _R7_             ; FIND_SOURCE
                        <1>
                        <1> ;-----      MEDIUM RES UP
000025B5 D0E2           <1>       SAL    DL, 1               ; # COLUMNS * 2, SINCE 2 BYTES/CHAR
000025B7 66D1E7         <1>       SAL    DI, 1               ; OFFSET *2 SINCE 2 BYTES/CHAR
                        <1>
                        <1> ;-----      DETERMINE THE SOURCE ADDRESS IN THE BUFFER
                        <1> _R7_:                                   ; FIND_SOURCE
000025BA 81C700800B00   <1>       add    edi, 0B8000h
000025C0 C0E702         <1>       sal    bh, 2               ; multiply number of lines by 4
000025C3 7431           <1>         JZ      short _R11             ; IF ZERO, THEN BLANK ENTIRE FIELD
000025C5 B050           <1>       MOV    AL, 80              ; 80 BYTES/ROW
000025C7 F6E7           <1>       mul    bh                  ; determine offset to source
000025C9 0FB7F0         <1>       movzx  esi, ax                  ; offset to source
000025CC 01FE           <1>       add    eSI, eDI            ; SET UP SOURCE
000025CE 88F4           <1>       MOV    AH, DH              ; NUMBER OF ROWS IN FIELD
000025D0 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
000025D2 E813040000     <1>         CALL    _R17                   ; MOVE ONE ROW
000025D7 6681EEB01F     <1>       SUB    SI, 2000h-80        ; MOVE TO NEXT ROW
000025DC 6681EFB01F     <1>       SUB    DI, 2000h-80
000025E1 FECC           <1>       DEC    AH                  ; NUMBER OF ROWS TO MOVE
000025E3 75ED           <1>         JNZ     short _R8              ; CONTINUE TILL ALL MOVED
                        <1>
                        <1> ;-----      FILL IN THE VACATED LINE(S)
                        <1> _R9:                                    ; CLEAR ENTRY
000025E5 88D8           <1>       mov    al, bl              ; attribute to fill with
                        <1> _R10_:
000025E7 E81A040000     <1>         CALL    _R18                   ; CLEAR THAT ROW
000025EC 6681EFB01F     <1>       SUB    DI, 2000h-80        ; POINT TO NEXT LINE
000025F1 FECF           <1>       dec    bh                  ; number of lines to fill
000025F3 75F2           <1>         JNZ     short _R10_            ; CLEAR LOOP
000025F5 C3             <1>       retn                       ; EVERYYHING DONE
                        <1>
                        <1> _R11:                                   ; BLANK_FIELD
000025F6 88F7           <1>       mov    bh, dh              ; set blank count to everything in field
000025F8 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'
                        <1>       ;
                        <1>
                        <1>       ; cl = upper left column
                        <1>       ; ch = upper left row
                        <1>       ; dl = lower rigth column
                        <1>       ; dh = lower right row
                        <1>       ;
                        <1>       ; al = line count (AL=0 means blank entire fields)
                        <1>       ; bl = fill value for blanked lines
                        <1>       ; bh = unused
                        <1>       ;
                        <1>       ; ah = [CRT_MODE], current video mode
                        <1>
000025FA 88C7           <1>       mov    bh, al ; 31/07/2016
000025FC BE[DE6F0000]   <1>       mov    esi, vga_g_modes
00002601 89F7           <1>       mov    edi, esi
00002603 83C708         <1>       add    edi, vga_g_mode_count
```

```
4052                                    <1> vga_g_up_0:
4053 00002606 AC                        <1>         lodsb
4054 00002607 38E0                      <1>         cmp     al, ah ; [CRT_MODE]
4055 00002609 7405                      <1>         je      short vga_g_up_1
4056 0000260B 39FE                      <1>         cmp     esi, edi
4057 0000260D 72F7                      <1>         jb      short vga_g_up_0
4058                                    <1>         ;xor    bh, bh ; 31/07/2016)
4059 0000260F C3                        <1>         retn    ; nothing to do
4060                                    <1> vga_g_up_1:
4061 00002610 88F8                      <1>         mov     al, bh ; 31/07/2016
4062 00002612 83C64F                    <1>         add     esi, vga_g_memmodel - (vga_g_modes + 1)
4063                                    <1>         ; [ESI] = VGA memory model number (LINEAR8, PLANAR4, PLANAR1)
4064                                    <1>
4065                                    <1>         ; if(rlr>=nbrows)rlr=nbrows-1;
4066                                    <1>         ; if(clr>=nbcols)clr=nbcols-1;
4067                                    <1>         ; if(nblines>nbrows)nblines=0;
4068                                    <1>         ; cols=clr-cul+1;
4069                                    <1>
4070 00002615 3A35[C26F0000]            <1>         cmp     dh, [VGA_ROWS]
4071 0000261B 7208                      <1>         jb      short vga_g_up_2
4072 0000261D 8A35[C26F0000]            <1>         mov     dh, [VGA_ROWS]
4073 00002623 FECE                      <1>         dec     dh
4074                                    <1> vga_g_up_2:
4075 00002625 3A15[BC6F0000]            <1>         cmp     dl, [CRT_COLS]  ; = [VGA_COLS]
4076 0000262B 7208                      <1>         jb      short vga_g_up_3
4077 0000262D 8A15[BC6F0000]            <1>         mov     dl, [CRT_COLS]
4078 00002633 FECA                      <1>         dec     dl
4079                                    <1> vga_g_up_3:
4080 00002635 3A05[C26F0000]            <1>         cmp     al, [VGA_ROWS]
4081 0000263B 7602                      <1>         jna     short vga_g_up_4
4082 0000263D 28C0                      <1>         sub     al, al ; 0
4083                                    <1> vga_g_up_4:
4084 0000263F 88D7                      <1>         mov     bh, dl ; clr
4085 00002641 28CF                      <1>         sub     bh, cl ; cul
4086 00002643 FEC7                      <1>         inc     bh ; cols = clr-cul+1
4087                                    <1>
4088 00002645 20C0                      <1>         and     al, al ; nblines = 0
4089 00002647 755D                      <1>         jnz     short vga_g_up_6
4090 00002649 20ED                      <1>         and     ch, ch ; rul = 0
4091 0000264B 7559                      <1>         jnz     short vga_g_up_6
4092 0000264D 20C9                      <1>         and     cl, cl ; cul = 0
4093 0000264F 7555                      <1>         jnz     short vga_g_up_6
4094                                    <1>
4095 00002651 6650                      <1>         push    ax
4096 00002653 A0[C26F0000]              <1>         mov     al, [VGA_ROWS]
4097 00002658 FEC8                      <1>         dec     al
4098 0000265A 38C6                      <1>         cmp     dh, al ; rlr = nbrows-1
4099 0000265C 7546                      <1>         jne     short vga_g_up_5
4100 0000265E A0[BC6F0000]              <1>           mov     al, [CRT_COLS]  ; = VGA_COLS
4101 00002663 FEC8                      <1>         dec     al
4102 00002665 38C2                      <1>         cmp     dl, al ; clr = nbcols-1
4103 00002667 753B                      <1>         jne     short vga_g_up_5
4104 00002669 6658                      <1>         pop     ax
4105                                    <1>
4106 0000266B 66B80502                  <1>         mov     ax, 0205h
4107 0000266F 66BACE03                  <1>         mov     dx, 3CEh ; VGAREG_GRDC_ADDRESS
4108 00002673 66EF                      <1>         out     dx, ax
4109 00002675 A0[C26F0000]              <1>         mov     al, [VGA_ROWS]
4110 0000267A 8A25[BC6F0000]            <1>         mov     ah, [CRT_COLS] ; = [VGA_COLS]
4111 00002680 F6E4                      <1>         mul     ah
4112 00002682 0FB7D0                    <1>         movzx   edx, ax
4113                                    <1>         ; 08/08/2016
4114 00002685 0FB605[BE6F0000]          <1>         movzx   eax, byte [CHAR_HEIGHT]
4115 0000268C F7E2                      <1>         mul     edx
4116                                    <1>         ; eax = byte count
4117 0000268E 89C1                      <1>         mov     ecx, eax
4118                                    <1>         ;; 07/08/2016
4119                                    <1>         ;shl    dx, 3 ; * 8 ; * [CHAR_HEIGHT]
4120                                    <1>         ;mov    ecx, edx
4121 00002690 88D8                      <1>         mov     al, bl ; fill value for blanked lines
4122 00002692 BF00000A00                <1>         mov     edi, 0A0000h
4123 00002697 F3AA                      <1>         rep     stosb
4124                                    <1>
4125 00002699 66B80500                  <1>         mov     ax, 5
4126 0000269D 66BACE03                  <1>         mov     dx, 3CEh ; VGAREG_GRDC_ADDRESS
4127 000026A1 66EF                      <1>         out     dx, ax ; 0005h
4128                                    <1>
4129 000026A3 C3                        <1>         retn
4130                                    <1>
4131                                    <1> vga_g_up_5:
4132 000026A4 6658                      <1>         pop     ax
4133                                    <1>
4134                                    <1> vga_g_up_6:
4135                                    <1>         ; [ESI] = VGA memory model number for current video mode
4136                                    <1>           ;
4137                                    <1>           ; LINEAR8 equ 5
4138                                    <1>           ; PLANAR4 equ 4
4139                                    <1>           ; PLANAR1 equ 3
4140                                    <1>
4141 000026A6 803E04                   <1>         cmp     byte [esi], PLANAR4
4142 000026A9 7424                      <1>         je      short vga_g_up_planar
4143 000026AB 803E03                    <1>         cmp     byte [esi], PLANAR1
4144 000026AE 741F                      <1>         je      short vga_g_up_planar
4145                                    <1> vga_g_up_linear8:
4146                                    <1>         ; 07/07/2016 (TEMPORARY)
4147                                    <1>         ;
4148                                    <1>         ; cl = upper left column ; cul
4149                                    <1>         ; ch = upper left row ; rul
4150                                    <1>         ; dl = lower rigth column ; clr
4151                                    <1>         ; dh = lower right row ; rlr
4152                                    <1>
4153                                    <1> vga_g_up_l0:
4154                                    <1>         ;{for(i=rul;i<=rlr;i++)
4155                                    <1>         ; if((i+nblines>rlr)||(nblines==0))
4156 000026B0 08C0                     <1>         or      al, al
```

```
4157 000026B2 7414                <1>       jz     short vga_g_up_l2
4158 000026B4 88C4                <1>       mov    ah, al
4159 000026B6 00EC                <1>       add    ah, ch ; i+nblines
4160                              <1>       ;jc    short vga_g_up_l2
4161 000026B8 38F4                <1>       cmp    ah, dh
4162 000026BA 770C                <1>       ja     short vga_g_up_l2
4163                              <1>       ; else
4164                              <1>       ;  vgamem_copy_pl4(cul,i+nblines,i,cols,nbcols,cheight);
4165 000026BC E8F2000000          <1>       call   vgamem_copy_l8
4166                              <1> vga_g_up_l1:
4167 000026C1 FEC5                <1>       inc    ch
4168 000026C3 38F5                <1>       cmp    ch, dh
4169 000026C5 76E9                <1>       jna    short vga_g_up_l0
4170 000026C7 C3                  <1>       retn
4171                              <1> vga_g_up_l2:
4172                              <1>       ; vgamem_fill_pl4(cul,i,cols,nbcols,cheight,attr);
4173 000026C8 E850010000          <1>       call   vgamem_fill_l8
4174 000026CD EBF2                <1>       jmp    short vga_g_up_l1
4175                              <1>
4176                              <1> vga_g_up_planar:
4177                              <1>       ; cl = upper left column ; cul
4178                              <1>       ; ch = upper left row ; rul
4179                              <1>       ; dl = lower rigth column ; clr
4180                              <1>       ; dh = lower right row ; rlr
4181                              <1> vga_g_up_pl0:
4182                              <1>       ;{for(i=rul;i<=rlr;i++)
4183                              <1>       ; if((i+nblines>rlr)||(nblines==0))
4184 000026CF 20C0                <1>       and    al, al
4185 000026D1 7414                <1>       jz     short vga_g_up_pl2
4186 000026D3 88C4                <1>       mov    ah, al
4187 000026D5 00EC                <1>       add    ah, ch ; i+nblines
4188                              <1>       ;jc    short vga_g_up_pl2
4189 000026D7 38F4                <1>       cmp    ah, dh
4190 000026D9 770C                <1>       ja     short vga_g_up_pl2
4191                              <1>       ; else
4192                              <1>       ;  vgamem_copy_pl4(cul,i+nblines,i,cols,nbcols,cheight);
4193 000026DB E80E000000          <1>       call   vgamem_copy_pl4
4194                              <1> vga_g_up_pl1:
4195 000026E0 FEC5                <1>       inc    ch
4196 000026E2 38F5                <1>       cmp    ch, dh
4197 000026E4 76E9                <1>       jna    short vga_g_up_pl0
4198 000026E6 C3                  <1>       retn
4199                              <1> vga_g_up_pl2:
4200                              <1>       ; vgamem_fill_pl4(cul,i,cols,nbcols,cheight,attr);
4201 000026E7 E870000000          <1>       call   vgamem_fill_pl4
4202 000026EC EBF2                <1>       jmp    short vga_g_up_pl1
4203                              <1>
4204                              <1> vgamem_copy_pl4:
4205                              <1>       ; 08/08/2016
4206                              <1>       ; 07/08/2016
4207                              <1>       ; 07/07/2016 - TRDOS 386 (TRDOS v2.0)
4208                              <1>       ;
4209                              <1>       ; derived from 'Plex86/Bochs VGABios' source code
4210                              <1>       ; vgabios-0.7a (2011)
4211                              <1>       ; by the LGPL VGABios developers Team (2001-2008)
4212                              <1>       ; 'vgabios.c', 'vgamem_copy_pl4'
4213                              <1>       ;
4214                              <1>       ; vgamem_copy_pl4(xstart,ysrc,ydest,cols,nbcols,cheight)
4215                              <1>       ; cl = xstart, ah = ysrc (i+nblines), ch = ydest (i),
4216                              <1>       ; bh = cols, [CRT_COLS] = nbcols, [CHAR_HEIGHT] = cheight
4217                              <1>
4218                              <1>       ; src=ysrc*cheight*nbcols+xstart;
4219                              <1>       ; dest=ydest*cheight*nbcols+xstart;
4220                              <1>
4221 000026EE 52                  <1>       push   edx
4222 000026EF 50                  <1>       push   eax
4223                              <1>
4224                              <1>       ; outw(VGAREG_GRDC_ADDRESS, 0x0105)
4225 000026F0 66B80501            <1>       mov    ax, 0105h
4226 000026F4 66BACE03            <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
4227 000026F8 66EF                <1>       out    dx, ax
4228                              <1>
4229                              <1>       ; 07/08/2016
4230                              <1>       ;mov     ah, [esp+1]
4231                              <1>       ;movzx edx, ah ; ysrc
4232 000026FA 0FB6542401          <1>       movzx  edx, byte [esp+1]
4233                              <1>       ; 08/08/2016
4234 000026FF 0FB605[BE6F0000]    <1>       movzx  eax, byte [CHAR_HEIGHT]
4235 00002706 8A25[BC6F0000]      <1>       mov    ah, [CRT_COLS] ; nbcols
4236 0000270C F6E4                <1>       mul    ah
4237                              <1>       ;; 07/08/2016
4238                              <1>       ;movzx eax, byte [CRT_COLS]
4239                              <1>       ;shl   ax, 3 ; * 8 ; * [CHAR_HEIGHT]
4240 0000270E 50                  <1>       push   eax ; cheight * nbcols
4241 0000270F F7E2                <1>       mul    edx ; * ysrc
4242                              <1>       ; eax = ysrc * cheight * nbcols
4243                              <1>       ; edx = 0
4244 00002711 88CA                <1>       mov    dl, cl ; edx = xstart
4245 00002713 01D0                <1>       add    eax, edx
4246 00002715 89C6                <1>       mov    esi, eax ; src
4247 00002717 88EA                <1>       mov    dl, ch ; ydest
4248 00002719 58                  <1>       pop    eax ; cheight * nbcols
4249 0000271A F7E2                <1>       mul    edx
4250                              <1>       ; eax = ydest * cheight * nbcols
4251 0000271C 88CA                <1>       mov    dl, cl ; edx = xstart
4252 0000271E 01D0                <1>       add    eax, edx
4253 00002720 89C7                <1>       mov    edi, eax ; dest
4254                              <1>       ; esi = src
4255                              <1>       ; edi = dest
4256                              <1>       ; for(i=0;i<cheight;i++)
4257                              <1>       ; {
4258                              <1>       ;  memcpyb(0xa000,dest+i*nbcols,0xa000,src+i*nbcols,cols);
4259                              <1>       ; }
4260 00002722 51                  <1>       push   ecx
4261 00002723 B900000A00          <1>       mov    ecx, 0A0000h
```

```
4262 00002728 01CE                <1>       add    esi, ecx
4263 0000272A 01CF                <1>       add    edi, ecx
4264                              <1>       ; 08/08/2016
4265 0000272C 8A35[BE6F0000]      <1>       mov    dh, [CHAR_HEIGHT]
4266                              <1>       ;; 07/08/2016
4267                              <1>       ;mov   dh, 8 ; 07/08/2016
4268 00002732 28D2                <1>       sub    dl, dl ; i
4269                              <1> vgamem_copy_pl4_0:
4270 00002734 56                  <1>       push   esi
4271 00002735 57                  <1>       push   edi
4272 00002736 0FB605[BC6F0000]    <1>       movzx  eax, byte [CRT_COLS]
4273 0000273D F6E2                <1>       mul    dl
4274                              <1>       ; eax = i * nbcols
4275 0000273F 01C7                <1>       add    edi, eax ; dest+i*nbcols
4276 00002741 01C6                <1>       add    esi, eax
4277 00002743 0FB6CF              <1>       movzx  ecx, bh ; cols
4278 00002746 F3A4                <1>       rep    movsb
4279 00002748 5F                  <1>       pop    edi
4280 00002749 5E                  <1>       pop    esi
4281 0000274A FECE                <1>       dec    dh
4282 0000274C 75E6                <1>       jnz    short vgamem_copy_pl4_0
4283                              <1> vgamem_copy_pl4_1:
4284 0000274E 59                  <1>       pop    ecx
4285                              <1>
4286                              <1>       ; outw(VGAREG_GRDC_ADDRESS, 0x0005);
4287 0000274F 66B80500            <1>       mov    ax, 0005h
4288 00002753 66BACE03            <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
4289 00002757 66EF                <1>       out    dx, ax
4290                              <1>
4291 00002759 58                  <1>       pop    eax
4292 0000275A 5A                  <1>       pop    edx
4293                              <1>
4294 0000275B C3                  <1>       retn
4295                              <1>
4296                              <1> vgamem_fill_pl4:
4297                              <1>       ; 08/08/2016
4298                              <1>       ; 07/08/2016
4299                              <1>       ; 04/08/2016
4300                              <1>       ; 07/07/2016 - TRDOS 386 (TRDOS v2.0)
4301                              <1>       ;
4302                              <1>       ; derived from 'Plex86/Bochs VGABios' source code
4303                              <1>       ; vgabios-0.7a (2011)
4304                              <1>       ; by the LGPL VGABios developers Team (2001-2008)
4305                              <1>       ; 'vgabios.c', 'vgamem_fill_pl4'
4306                              <1>       ;
4307                              <1>       ; vgamem_fill_pl4(xstart,ystart,cols,nbcols,cheight,attr)
4308                              <1>       ; cl = xstart, edi = ch = ystart, bh = cols,
4309                              <1>       ; [CRT_COLS] = nbcols, [CHAR_HEIGHT] = cheight, attr = 0
4310                              <1>
4311                              <1>       ; dest=ystart*cheight*nbcols+xstart;
4312 0000275C 52                  <1>       push   edx
4313 0000275D 50                  <1>       push   eax
4314                              <1>
4315                              <1>       ; outw(VGAREG_GRDC_ADDRESS, 0x0205)
4316 0000275E 66B80502            <1>       mov    ax, 0205h
4317 00002762 66BACE03            <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
4318 00002766 66EF                <1>       out    dx, ax
4319                              <1>
4320                              <1>              ; 08/08/2016
4321 00002768 0FB605[BE6F0000]    <1>       movzx   eax, byte [CHAR_HEIGHT]
4322 0000276F F6E5                <1>       mul    ch
4323                              <1>       ;; 07/08/2016
4324                              <1>       ;movzx eax, ch
4325                              <1>       ;shl   ax, 3 ; * 8 ; * [CHAR_HEIGHT]
4326 00002771 0FB615[BC6F0000]    <1>       movzx  edx, byte [CRT_COLS] ; = [VGA_COLS]
4327 00002778 F7E2                <1>       mul    edx
4328                              <1>       ; edx  = 0
4329 0000277A 88CA                <1>       mov    dl, cl
4330 0000277C 01D0                <1>       add    eax, edx
4331 0000277E 89C7                <1>       mov    edi, eax
4332                              <1>       ; edi = dest
4333                              <1>       ; for(i=0;i<cheight;i++)
4334                              <1>       ; {
4335                              <1>       ;  memsetb(0xa000,dest+i*nbcols,attr,cols);
4336                              <1>       ; }
4337 00002780 81C700000A00        <1>       add    edi, 0A0000h
4338 00002786 51                  <1>       push   ecx
4339                              <1>       ; 08/08/2016
4340 00002787 8A35[BE6F0000]      <1>       mov    dh, [CHAR_HEIGHT]
4341                              <1>       ;; 07/08/2016
4342                              <1>       ;mov   dh, 8 ; 07/08/2016
4343 0000278D 28D2                <1>       sub    dl, dl ; i
4344                              <1> vgamem_fill_pl4_0:
4345 0000278F 57                  <1>       push   edi
4346 00002790 0FB605[BC6F0000]    <1>       movzx  eax, byte [CRT_COLS]
4347 00002797 F6E2                <1>       mul    dl
4348                              <1>       ; eax = i * nbcols
4349 00002799 01C7                <1>       add    edi, eax ; dest+i*nbcols
4350 0000279B 88D8                <1>       mov    al, bl ; attr ; 04/08/2016
4351 0000279D 0FB6CF              <1>       movzx  ecx, bh ; cols
4352 000027A0 F3AA                <1>       rep    stosb
4353 000027A2 5F                  <1>       pop    edi
4354 000027A3 75EA                <1>       jnz    short vgamem_fill_pl4_0
4355                              <1> vgamem_fill_pl4_1:
4356 000027A5 59                  <1>       pop    ecx
4357                              <1>
4358                              <1>       ; outw(VGAREG_GRDC_ADDRESS, 0x0005);
4359 000027A6 66B80500            <1>       mov    ax, 0005h
4360 000027AA 66BACE03            <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
4361 000027AE 66EF                <1>       out    dx, ax
4362                              <1>
4363 000027B0 58                  <1>       pop    eax
4364 000027B1 5A                  <1>       pop    edx
4365                              <1>
4366 000027B2 C3                  <1>       retn
```

```
4367                                    <1>
4368                                    <1> vgamem_copy_l8:
4369                                    <1>         ; 08/08/2016
4370                                    <1>         ; 07/08/2016
4371                                    <1>         ; 06/08/2016
4372                                    <1>         ; 07/07/2016 - TRDOS 386 (TRDOS v2.0)
4373                                    <1>         ;
4374                                    <1>         ; TEMPORARY
4375                                    <1>         ;
4376                                    <1>         ; derived from 'Plex86/Bochs VGABios' source code
4377                                    <1>         ; vgabios-0.7a (2011)
4378                                    <1>         ; by the LGPL VGABios developers Team (2001-2008)
4379                                    <1>         ; 'vgabios.c', 'vgamem_copy_pl4'
4380                                    <1>         ;
4381                                    <1>         ; vgamem_copy_pl4(xstart,ysrc,ydest,cols,nbcols,cheight)
4382                                    <1>         ; cl = xstart, ah = ysrc (i+nblines), ch = ydest (i),
4383                                    <1>         ; bh = cols, [CRT_COLS] = nbcols, [CHAR_HEIGHT] = cheight
4384                                    <1>
4385                                    <1>         ; src=ysrc*cheight*nbcols+xstart;
4386                                    <1>         ; dest=ydest*cheight*nbcols+xstart;
4387                                    <1>
4388 000027B3 52                     <1>         push   edx
4389 000027B4 50                     <1>         push   eax
4390                                    <1>
4391                                    <1>         ; outw(VGAREG_GRDC_ADDRESS, 0x0105)
4392                                    <1>         ;mov   ax, 0105h
4393                                    <1>         ;mov   dx, 3CEh ; VGAREG_GRDC_ADDRESS
4394                                    <1>         ;out   dx, ax
4395                                    <1>
4396                                    <1>         ;mov   ah, [esp+1]
4397                                    <1>
4398 000027B5 0FB6D4                  <1>         movzx  edx, ah ; ysrc
4399                                    <1>         ; 08/08/2016
4400 000027B8 0FB605[BE6F0000]         <1>         movzx  eax, byte [CHAR_HEIGHT]
4401 000027BF 8A25[BC6F0000]           <1>         mov    ah, [CRT_COLS] ; nbcols
4402 000027C5 F6E4                     <1>         mul    ah
4403                                    <1>         ;; 07/08/2016
4404                                    <1>         ;movzx eax, byte [CRT_COLS]
4405                                    <1>         ;shl   ax, 3 ; * 8 ; * [CHAR_HEIGHT]
4406 000027C7 50                     <1>         push   eax ; cheight * nbcols
4407 000027C8 F7E2                     <1>         mul    edx ; * ysrc
4408                                    <1>         ; eax = ysrc * cheight * nbcols
4409                                    <1>         ; edx = 0
4410 000027CA 88CA                     <1>         mov    dl, cl ; edx = xstart
4411 000027CC 01D0                     <1>         add    eax, edx
4412 000027CE 89C6                     <1>         mov    esi, eax ; src
4413 000027D0 66C1E603                 <1>         shl    si, 3 ; * 8 ; 06/08/2016
4414 000027D4 88EA                     <1>         mov    dl, ch ; ydest
4415 000027D6 58                     <1>         pop    eax ; cheight * nbcols
4416 000027D7 F7E2                     <1>         mul    edx
4417                                    <1>         ; eax = ydest * cheight * nbcols
4418 000027D9 88CA                     <1>         mov    dl, cl ; edx = xstart
4419 000027DB 01D0                     <1>         add    eax, edx
4420 000027DD 89C7                     <1>         mov    edi, eax ; dest
4421 000027DF 66C1E703                 <1>         shl    di, 3 ; * 8 ; 06/08/2016
4422                                    <1>         ; esi = src
4423                                    <1>         ; edi = dest
4424                                    <1>         ; for(i=0;i<cheight;i++)
4425                                    <1>         ; {
4426                                    <1>         ;  memcpyb(0xa000,dest+i*nbcols,0xa000,src+i*nbcols,cols);
4427                                    <1>         ; }
4428 000027E3 51                     <1>         push   ecx
4429 000027E4 B900000A00               <1>         mov    ecx, 0A0000h
4430 000027E9 01CE                     <1>         add    esi, ecx
4431 000027EB 01CF                     <1>         add    edi, ecx
4432                                    <1>         ; 08/08/2016
4433 000027ED 8A35[BE6F0000]           <1>         mov    dh, [CHAR_HEIGHT]
4434                                    <1>         ;; 07/08/2016
4435                                    <1>         ;mov   dh, 8 ; 07/08/2016
4436 000027F3 28D2                     <1>         sub    dl, dl ; i
4437                                    <1> vgamem_copy_l8_0:
4438 000027F5 56                     <1>         push   esi
4439 000027F6 57                     <1>         push   edi
4440 000027F7 0FB605[BC6F0000]         <1>         movzx  eax, byte [CRT_COLS]
4441 000027FE F6E2                     <1>         mul    dl
4442                                    <1>         ; eax = i * nbcols
4443 00002800 66C1E003                 <1>         shl    ax, 3 ; * 8 ; 06/08/2016
4444 00002804 01C7                     <1>         add    edi, eax ; dest+i*nbcols
4445 00002806 01C6                     <1>         add    esi, eax
4446 00002808 0FB6CF                   <1>         movzx  ecx, bh ; cols
4447 0000280B 66C1E103                 <1>         shl    cx, 3 ; * 8 ; 06/08/2016
4448 0000280F F3A4                     <1>         rep    movsb
4449 00002811 5F                     <1>         pop    edi
4450 00002812 5E                     <1>         pop    esi
4451 00002813 FEC2                     <1>         inc    dl ; 06/08/2016
4452 00002815 FECE                     <1>         dec    dh
4453 00002817 75DC                     <1>         jnz    short vgamem_copy_l8_0
4454                                    <1> vgamem_copy_l8_1:
4455 00002819 59                     <1>         pop    ecx
4456                                    <1>
4457                                    <1>         ;; outw(VGAREG_GRDC_ADDRESS, 0x0005);
4458                                    <1>         ;mov   ax, 0005h
4459                                    <1>         ;mov   dx, 3CEh ; VGAREG_GRDC_ADDRESS
4460                                    <1>         ;out   dx, ax
4461                                    <1>
4462 0000281A 58                     <1>         pop    eax
4463 0000281B 5A                     <1>         pop    edx
4464                                    <1>
4465 0000281C C3                     <1>         retn
4466                                    <1>
4467                                    <1> vgamem_fill_l8:
4468                                    <1>         ; 08/08/2016
4469                                    <1>         ; 07/08/2016
4470                                    <1>         ; 06/08/2016
4471                                    <1>         ; 04/08/2016
```

```
4472                                    <1>       ; 07/07/2016 - TRDOS 386 (TRDOS v2.0)
4473                                    <1>       ;
4474                                    <1>       ; TEMPORARY
4475                                    <1>       ;
4476                                    <1>       ; derived from 'Plex86/Bochs VGABios' source code
4477                                    <1>       ; vgabios-0.7a (2011)
4478                                    <1>       ; by the LGPL VGABios developers Team (2001-2008)
4479                                    <1>       ; 'vgabios.c', 'vgamem_fill_pl4'
4480                                    <1>       ;
4481                                    <1>       ; vgamem_fill_pl4(xstart,ystart,cols,nbcols,cheight,attr)
4482                                    <1>       ; cl = xstart, edi = ch = ystart, bh = cols,
4483                                    <1>       ; [CRT_COLS] = nbcols, [CHAR_HEIGHT] = cheight, attr = 0
4484                                    <1>
4485                                    <1>       ; dest=ystart*cheight*nbcols+xstart;
4486 0000281D 52                      <1>       push   edx
4487 0000281E 50                      <1>       push   eax
4488                                    <1>
4489                                    <1>       ;; outw(VGAREG_GRDC_ADDRESS, 0x0205)
4490                                    <1>       ;mov   ax, 0205h
4491                                    <1>       ;mov   dx, 3CEh ; VGAREG_GRDC_ADDRESS
4492                                    <1>       ;out   dx, ax
4493                                    <1>
4494                                    <1>         ; 08/08/2016
4495 0000281F 0FB605[BE6F0000]         <1>       movzx   eax, byte [CHAR_HEIGHT]
4496 00002826 F6E5                     <1>       mul    ch
4497                                    <1>       ;; 07/08/2016
4498                                    <1>       ;movzx eax, ch
4499                                    <1>       ;shl   ax, 3 ; * 8 ; * [CHAR_HEIGHT]
4500 00002828 0FB615[BC6F0000]         <1>       movzx  edx, byte [CRT_COLS] ; = [VGA_COLS]
4501 0000282F F7E2                     <1>       mul    edx
4502                                    <1>       ; edx  = 0
4503 00002831 88CA                     <1>       mov    dl, cl
4504 00002833 01D0                     <1>       add    eax, edx
4505 00002835 89C7                     <1>       mov    edi, eax
4506 00002837 66C1E703                 <1>       shl    di, 3 ; * 8 ; 06/08/2016
4507                                    <1>       ; edi = dest
4508                                    <1>       ; for(i=0;i<cheight;i++)
4509                                    <1>       ; {
4510                                    <1>       ;  memsetb(0xa000,dest+i*nbcols,attr,cols);
4511                                    <1>       ; }
4512 0000283B 81C700000A00              <1>       add    edi, 0A0000h
4513 00002841 51                       <1>       push   ecx
4514                                    <1>       ; 08/08/2016
4515 00002842 8A35[BE6F0000]           <1>       mov    dh, [CHAR_HEIGHT]
4516                                    <1>       ;; 07/08/2016
4517                                    <1>       ;mov   dh, 8 ; 07/08/2016
4518 00002848 28D2                     <1>       sub    dl, dl ; i
4519                                    <1> vgamem_fill_l8_0:
4520 0000284A 57                       <1>       push   edi
4521 0000284B 0FB605[BC6F0000]         <1>       movzx  eax, byte [CRT_COLS]
4522 00002852 F6E2                     <1>       mul    dl
4523                                    <1>       ; eax = i * nbcols
4524 00002854 66C1E003                 <1>       shl    ax, 3 ; * 8 ; 06/08/2016
4525 00002858 01C7                     <1>       add    edi, eax ; dest+i*nbcols
4526 0000285A 88D8                     <1>       mov    al, bl ; attr ; 04/08/2016
4527 0000285C 0FB6CF                   <1>       movzx  ecx, bh ; cols
4528 0000285F 66C1E103                 <1>       shl    cx, 3 ; * 8 ; 06/08/2016
4529 00002863 F3AA                     <1>       rep    stosb
4530 00002865 5F                       <1>       pop    edi
4531 00002866 FEC2                     <1>       inc    dl ; 06/08/2016
4532 00002868 FECE                     <1>       dec    dh
4533 0000286A 75DE                     <1>       jnz    short vgamem_fill_l8_0
4534                                    <1> vgamem_fill_l8_1:
4535 0000286C 59                       <1>       pop    ecx
4536                                    <1>
4537                                    <1>       ;; outw(VGAREG_GRDC_ADDRESS, 0x0005);
4538                                    <1>       ;mov   ax, 0005h
4539                                    <1>       ;mov   dx, 3CEh ; VGAREG_GRDC_ADDRESS
4540                                    <1>       ;out   dx, ax
4541                                    <1>
4542 0000286D 58                       <1>       pop    eax
4543 0000286E 5A                       <1>       pop    edx
4544                                    <1>
4545 0000286F C3                       <1>       retn
4546                                    <1>
4547                                    <1> vga_graphics_down:
4548                                    <1>       ; 08/08/2016
4549                                    <1>       ; 07/08/2016
4550                                    <1>       ; 31/07/2016
4551                                    <1>       ; 07/07/2016 - TRDOS 386 (TRDOS v2.0)
4552                                    <1>       ;
4553                                    <1>       ; derived from 'Plex86/Bochs VGABios' source code
4554                                    <1>       ; vgabios-0.7a (2011)
4555                                    <1>       ; by the LGPL VGABios developers Team (2001-2008)
4556                                    <1>       ; 'vgabios.c', 'biosfn_scroll'
4557                                    <1>       ;
4558                                    <1>
4559                                    <1>       ; cl = upper left column
4560                                    <1>       ; ch = upper left row
4561                                    <1>       ; dl = lower rigth column
4562                                    <1>       ; dh = lower right row
4563                                    <1>       ;
4564                                    <1>       ; al = line count (AL=0 means blank entire fields)
4565                                    <1>       ; bl = fill value for blanked lines
4566                                    <1>       ; bh = unused
4567                                    <1>       ;
4568                                    <1>       ; ah = [CRT_MODE], current video mode
4569                                    <1>
4570 00002870 FC                       <1>         cld     ; !!! Clear direction flag !!!
4571                                    <1>
4572 00002871 88C7                     <1>       mov    bh, al ; 31/07/2016
4573                                    <1>
4574 00002873 BE[D66F0000]             <1>       mov    esi, vga_modes
4575 00002878 89F7                     <1>       mov    edi, esi
4576 0000287A 83C710                   <1>       add    edi, vga_mode_count
```

```
4577                                    <1> vga_g_down_0:
4578 0000287D AC                        <1>         lodsb
4579 0000287E 38E0                      <1>         cmp     al, ah ; [CRT_MODE]
4580 00002880 7405                      <1>         je      short vga_g_down_1
4581 00002882 39FE                      <1>         cmp     esi, edi
4582 00002884 72F7                      <1>         jb      short vga_g_down_0
4583                                    <1>         ; xor  bh, bh ; 31/07/2016
4584 00002886 C3                        <1>         retn    ; nothing to do
4585                                    <1> vga_g_down_1:
4586 00002887 88F8                      <1>         mov     al, bh ; 31/07/2016
4587 00002889 83C64F                    <1>         add     esi, vga_memmodel - (vga_modes + 1)
4588                                    <1>         ; [ESI] = VGA memory model number (LINEAR8, PLANAR4, PLANAR1)
4589                                    <1>
4590                                    <1>         ; if(rlr>=nbrows)rlr=nbrows-1;
4591                                    <1>         ; if(clr>=nbcols)clr=nbcols-1;
4592                                    <1>         ; if(nblines>nbrows)nblines=0;
4593                                    <1>         ; cols=clr-cul+1;
4594                                    <1>
4595 0000288C 3A35[C26F0000]            <1>         cmp     dh, [VGA_ROWS]
4596 00002892 7208                      <1>         jb      short vga_g_down_2
4597 00002894 8A35[C26F0000]            <1>         mov     dh, [VGA_ROWS]
4598 0000289A FECE                      <1>         dec     dh
4599                                    <1> vga_g_down_2:
4600 0000289C 3A15[BC6F0000]            <1>         cmp     dl, [CRT_COLS]  ; = [VGA_COLS]
4601 000028A2 7208                      <1>         jb      short vga_g_down_3
4602 000028A4 8A15[BC6F0000]            <1>         mov     dl, [CRT_COLS]
4603 000028AA FECA                      <1>         dec     dl
4604                                    <1> vga_g_down_3:
4605 000028AC 3A05[C26F0000]            <1>         cmp     al, [VGA_ROWS]
4606 000028B2 7602                      <1>         jna     short vga_g_down_4
4607 000028B4 28C0                      <1>         sub     al, al ; 0
4608                                    <1> vga_g_down_4:
4609 000028B6 88F7                      <1>         mov     bh, dh ; clr
4610 000028B8 28CF                      <1>         sub     bh, cl ; cul
4611 000028BA FEC7                      <1>         inc     bh ; cols = clr-cul+1
4612                                    <1>
4613 000028BC 20C0                      <1>         and     al, al ; nblines = 0
4614 000028BE 755B                      <1>         jnz     short vga_g_down_6
4615 000028C0 20ED                      <1>         and     ch, ch ; rul = 0
4616 000028C2 7557                      <1>         jnz     short vga_g_down_6
4617 000028C4 20C9                      <1>         and     cl, cl ; cul = 0
4618 000028C6 7553                      <1>         jnz     short vga_g_down_6
4619                                    <1>
4620 000028C8 6650                      <1>         push    ax
4621 000028CA A0[C26F0000]              <1>         mov     al, [VGA_ROWS]
4622 000028CF FEC8                      <1>         dec     al
4623 000028D1 38C6                      <1>         cmp     dh, al ; rlr = nbrows-1
4624 000028D3 7544                      <1>         jne     short vga_g_down_5
4625 000028D5 A0[BC6F0000]              <1>           mov     al, [CRT_COLS]  ; = VGA_COLS
4626 000028DA FEC8                      <1>         dec     al
4627 000028DC 38C2                      <1>         cmp     dl, al ; clr = nbcols-1
4628 000028DE 7539                      <1>         jne     short vga_g_down_5
4629 000028E0 6658                      <1>         pop     ax
4630                                    <1>
4631 000028E2 66B80502                  <1>         mov     ax, 0205h
4632 000028E6 66BACE03                  <1>         mov     dx, 3CEh ; VGAREG_GRDC_ADDRESS
4633 000028EA 66EF                      <1>         out     dx, ax
4634 000028EC A0[C26F0000]              <1>         mov     al, [VGA_ROWS]
4635 000028F1 8A25[BC6F0000]            <1>         mov     ah, [CRT_COLS] ; = [VGA_COLS]
4636 000028F7 F6E4                      <1>         mul     ah
4637 000028F9 0FB7D0                    <1>         movzx  edx, ax
4638                                    <1>         ; 08/08/2016
4639 000028FC 0FB605[BE6F0000]          <1>         movzx  eax, byte [CHAR_HEIGHT]
4640 00002903 F7E2                      <1>         mul     edx
4641                                    <1>         ; eax = byte count
4642 00002905 89C1                      <1>         mov     ecx, eax
4643                                    <1>         ;; 07/08/2016
4644                                    <1>         ;shl   dx, 3 ; * 8 ; * [CHAR_HEIGHT]
4645                                    <1>         ;mov   ecx, edx
4646 00002907 88D8                      <1>         mov     al, bl ; fill value for blanked lines
4647 00002909 BF00000A00                <1>         mov     edi, 0A0000h
4648 0000290E F3AA                      <1>         rep     stosb
4649                                    <1>
4650 00002910 B005                      <1>         mov     al, 5
4651 00002912 66BACE03                  <1>         mov     dx, 3CEh ; VGAREG_GRDC_ADDRESS
4652 00002916 66EF                      <1>         out     dx, ax ; 0005h
4653                                    <1>
4654 00002918 C3                        <1>         retn
4655                                    <1>
4656                                    <1> vga_g_down_5:
4657 00002919 6658                      <1>         pop     ax
4658                                    <1>
4659                                    <1> vga_g_down_6:
4660                                    <1>         ; [ESI] = VGA memory model number for current video mode
4661                                    <1>           ;
4662                                    <1>           ; LINEAR8 equ 5
4663                                    <1>           ; PLANAR4 equ 4
4664                                    <1>           ; PLANAR1 equ 3
4665                                    <1>
4666 0000291B 803E04                    <1>         cmp     byte [esi], PLANAR4
4667 0000291E 742C                      <1>         je      short vga_g_down_planar
4668 00002920 803E03                    <1>         cmp     byte [esi], PLANAR1
4669 00002923 7427                      <1>         je      short vga_g_down_planar
4670                                    <1> vga_g_down_linear8:
4671                                    <1>         ; 07/07/2016 (TEMPORARY)
4672                                    <1>         ;
4673                                    <1>         ; cl = upper left column ; cul
4674                                    <1>         ; ch = upper left row ; rul
4675                                    <1>         ; dl = lower rigth column ; clr
4676                                    <1>         ; dh = lower right row ; rlr
4677                                    <1>
4678                                    <1> vga_g_down_l0:
4679                                    <1>         ;{for(i=rlr;i>=rul;i--)
4680                                    <1>           ; if((i<rul+nblines)||(nblines==0))
4681 00002925 08C0                      <1>         or      al, al
```

```
4682 00002927 741C              <1>       jz      short vga_g_down_l2
4683 00002929 88C4              <1>       mov     ah, al
4684 0000292B 00EC              <1>       add     ah, ch
4685                            <1>       ;jc     short vga_g_down_l2
4686 0000292D 86EE              <1>       xchg    ch, dh
4687 0000292F 38E5              <1>       cmp     ch, ah
4688 00002931 7212              <1>       jb      short vga_g_down_l2
4689 00002933 88EC              <1>       mov     ah, ch
4690 00002935 28C4              <1>       sub     ah, al ; ah = i - nblines
4691                            <1>       ; else
4692                            <1>       ; vgamem_copy_pl4(cul,i,i-nblines,cols,nbcols,cheight);
4693 00002937 E877FEFFFF        <1>       call    vgamem_copy_l8
4694                            <1> vga_g_down_l1:
4695 0000293C 86F5              <1>       xchg    dh, ch
4696 0000293E FECE              <1>       dec     dh
4697 00002940 38EE              <1>       cmp     dh, ch
4698 00002942 73E1              <1>       jnb     short vga_g_down_l0
4699 00002944 C3                <1>       retn
4700                            <1>
4701                            <1> vga_g_down_l2:
4702                            <1>       ; vgamem_fill_pl4(cul,i,cols,nbcols,cheight,attr);
4703 00002945 E8D3FEFFFF        <1>       call    vgamem_fill_l8
4704 0000294A EBF0              <1>       jmp     short vga_g_down_l1
4705                            <1>
4706                            <1> vga_g_down_planar:
4707                            <1>       ; cl = upper left column ; cul
4708                            <1>       ; ch = upper left row ; rul
4709                            <1>       ; dl = lower rigth column ; clr
4710                            <1>       ; dh = lower right row ; rlr
4711                            <1> vga_g_down_pl0:
4712                            <1>       ;{for(i=rlr;i>=rul;i--)
4713                            <1>         ; if((i<rul+nblines)||(nblines==0))
4714 0000294C 08C0              <1>       or      al, al
4715 0000294E 741C              <1>       jz      short vga_g_down_pl2
4716 00002950 88C4              <1>       mov     ah, al
4717 00002952 00EC              <1>       add     ah, ch
4718                            <1>       ;jc     short vga_g_down_pl2
4719 00002954 86EE              <1>       xchg    ch, dh
4720 00002956 38E5              <1>       cmp     ch, ah
4721 00002958 7212              <1>       jb      short vga_g_down_pl2
4722 0000295A 88EC              <1>       mov     ah, ch
4723 0000295C 28C4              <1>       sub     ah, al ; ah = i - nblines
4724                            <1>       ; else
4725                            <1>       ; vgamem_copy_pl4(cul,i,i-nblines,cols,nbcols,cheight);
4726 0000295E E88BFDFFFF        <1>       call    vgamem_copy_pl4
4727                            <1> vga_g_down_pl1:
4728 00002963 86F5              <1>       xchg    dh, ch
4729 00002965 FECE              <1>       dec     dh
4730 00002967 38EE              <1>       cmp     dh, ch
4731 00002969 73E1              <1>       jnb     short vga_g_down_pl0
4732 0000296B C3                <1>       retn
4733                            <1>
4734                            <1> vga_g_down_pl2:
4735                            <1>       ; vgamem_fill_pl4(cul,i,cols,nbcols,cheight,attr);
4736 0000296C E8EBFDFFFF        <1>       call    vgamem_fill_pl4
4737 00002971 EBF0              <1>       jmp     short vga_g_down_pl1
4738                            <1>
4739                            <1> ; 07/07/2016
4740                            <1> ; 27/06/2016 - TRDOS 386 (TRDOS v2.0)
4741                            <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
4742                            <1> ;--------------------------------------------------
4743                            <1> ; SCROLL DOWN
4744                            <1> ;  THIS ROUTINE SCROLLS DOWN THE INFORMATION ON THE CRT
4745                            <1> ; ENTRY --
4746                            <1> ;  CH,CL = UPPER LEFT CORNER OF REGION TO SCROLL
4747                            <1> ;  DH,DL = LOWER RIGHT CORNER OF REGION TO SCROLL
4748                            <1> ;   BOTH OF THE ABOVE ARE IN CHARACTER POSITIONS
4749                            <1> ;  BH = FILL VALUE FOR BLANKED LINES
4750                            <1> ;  AL = # LINES TO SCROLL (AL=0 MEANS BLANK THE ENTIRE FIELD)
4751                            <1> ;  DS = DATA SEGMENT
4752                            <1> ;  ES = REGEN SEGMENT
4753                            <1> ; EXIT --
4754                            <1> ;  NOTHING, THE SCREEN IS SCROLLED
4755                            <1> ;--------------------------------------------------
4756                            <1>
4757                            <1>       ; cl = upper left column
4758                            <1>       ; ch = upper left row
4759                            <1>       ; dl = lower rigth column
4760                            <1>       ; dh = lower right row
4761                            <1>       ;
4762                            <1>       ; al = line count (AL=0 means blank entire fields)
4763                            <1>       ; bl = fill value for blanked lines
4764                            <1>       ; bh = unused
4765                            <1>
4766                            <1> GRAPHICS_DOWN:
4767                            <1>       ; 07/07/2016
4768                            <1>       ;AH = Current video mode, [CRT_MODE]
4769                            <1>       ;STD                    ; SET DIRECTION
4770 00002973 80FC07            <1>       cmp    ah, 7
4771 00002976 0F87F4FEFFFF      <1>         ja      vga_graphics_down
4772                            <1>       ;je    _n0
4773                            <1>
4774 0000297C 88C7              <1>       MOV    bh, al             ; save line count in BH
4775 0000297E 6689D0            <1>       MOV    AX, DX             ; GET LOWER RIGHT POSITION INTO AX REG
4776                            <1>
4777                            <1> ;-----      USE CHARACTER SUBROUTINE FOR POSITIONING
4778                            <1> ;-----      ADDRESS RETURNED IS MULTIPLIED BY 2 FROM CORRECT VALUE
4779                            <1>
4780 00002981 E8F2010000        <1>       CALL   GRAPH_POSN
4781 00002986 0FB7F8            <1>       MOVzx  eDI, AX                  ; SAVE RESULT AS DESTINATION ADDRESS
4782                            <1>
4783                            <1> ;-----      DETERMINE SIZE OF WINDOW
4784                            <1>
4785 00002989 6629CA            <1>       SUB    DX, CX
4786 0000298C 6681C20101        <1>         ADD     DX, 101h               ; ADJUST VALUES
```

```
4787 00002991 C0E602              <1>       SAL    DH, 2              ; MULTIPLY ROWS BY 4 AT 8 VERT DOTS/CHAR
4788                              <1>                                  ; AND EVEN/ODD ROWS
4789                              <1>
4790                              <1> ;-----      DETERMINE CRT MODE
4791                              <1>
4792 00002994 803D[BA6F0000]06    <1>       CMP    byte [CRT_MODE], 6  ; TEST FOR MEDIUM RES
4793 0000299B 7307                <1>         JNC     short _R12             ; FIND_SOURCE_DOWN
4794                              <1>
4795                              <1> ;-----      MEDIUM RES DOWN
4796 0000299D D0E2                <1>       SAL    DL, 1              ; # COLUMNS * 2, SINCE 2 BYTES/CHAR
4797 0000299F 66D1E7              <1>       SAL    DI, 1              ; OFFSET *2 SINCE 2 BYTES/CHAR
4798 000029A2 6647                <1>       INC    DI                 ; POINT TO LAST BYTE
4799                              <1>
4800                              <1> ;-----      DETERMINE THE SOURCE ADDRESS IN THE BUFFER
4801                              <1>
4802                              <1> _R12:                                  ; FIND_SOURCE_DOWN
4803 000029A4 81C700800B00        <1>       add    edi, 0B8000h
4804 000029AA 6681C7F000          <1>       ADD    DI, 240                ; POINT TO LAST ROW OF PIXELS
4805 000029AF C0E702              <1>       sal    bh, 2              ; multiply number of lines by 4
4806 000029B2 74(06)              <1>       JZ     short 6                 ; IF ZERO, THEN BLANK ENTIRE FIELD
4807 000029B4 B050                <1>       MOV    AL, 80             ; 80 BYTES/ROW
4808 000029B6 F6E7                <1>       mul    bh                 ; determine offset to source
4809 000029B8 89FE                <1>       MOV    eSI, eDI           ; SET UP SOURCE
4810 000029BA 6629C6              <1>       SUB    SI, AX             ; SUBTRACT THE OFFSET
4811 000029BD 88F4                <1>       MOV    AH, DH             ; NUMBER OF ROWS IN FIELD
4812 000029BF 28FC                <1>       sub    ah, bh             ; determine number to move
4813                              <1>
4814                              <1> ;-----      LOOP THROUGH, MOVING ONE ROW AT A TIME, BOTH EVEN AND ODD FIELDS
4815                              <1>
4816                              <1> _R13:                                  ; ROW_LOOP_DOWN
4817 000029C1 E824000000          <1>         CALL    _R17                   ; MOVE ONE ROW
4818 000029C6 6681EE5020          <1>       SUB    SI, 2000h+80       ; MOVE TO NEXT ROW
4819 000029CB 6681EF5020          <1>       SUB    DI, 2000h+80
4820 000029D0 FECC                <1>       DEC    AH                 ; NUMBER OF ROWS TO MOVE
4821 000029D2 75ED                <1>         JNZ     short _R13             ; CONTINUE TILL ALL MOVED
4822                              <1>
4823                              <1> ;-----      FILL IN THE VACATED LINE(S)
4824                              <1> _R14:                                  ; CLEAR_ENTRY_DOWN
4825 000029D4 88D8                <1>       mov    al, bl             ; attribute to fill with
4826                              <1> _R15_:                                 ; CLEAR_LOOP_DOWN
4827 000029D6 E82B000000          <1>       CALL   _R18               ; CLEAR A ROW
4828 000029DB 6681EF5020          <1>       SUB    DI, 2000h+80       ; POINT TO NEXT LINE
4829 000029E0 FECF                <1>       dec    bh                 ; number of lines to fill
4830 000029E2 75F2                <1>         JNZ     short _R15_            ; CLEAR_LOOP_DOWN
4831                              <1>       ; 18/11/2020
4832 000029E4 FC                  <1>       CLD                       ; RESET THE DIRECTION FLAG
4833                              <1>
4834 000029E5 C3                  <1>       retn                      ; EVERYTHING DONE
4835                              <1>
4836                              <1> _R16:                                  ; BLANK_FIELD_DOWN
4837 000029E6 88F7                <1>       mov    bh, dh             ; set blank count to everything in field
4838 000029E8 EBEA                <1>         JMP     short _R14             ; CLEAR THE FIELD
4839                              <1>
4840                              <1> ; 27/06/2016 - TRDOS 386 (TRDOS v2.0)
4841                              <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
4842                              <1>
4843                              <1> ;-----      ROUTINE TO MOVE ONE ROW OF INFORMATION
4844                              <1>
4845                              <1> _R17:
4846 000029EA 0FB6CA              <1>       MOVzx  ecx, DL                 ; NUMBER OF BYTES IN THE ROW
4847 000029ED 56                  <1>       PUSH   eSI
4848 000029EE 57                  <1>       PUSH   eDI                ; SAVE POINTERS
4849 000029EF F3A4                <1>       REP    MOVSB              ; MOVE THE EVEN FIELD
4850 000029F1 5F                  <1>       POP    eDI
4851 000029F2 5E                  <1>       POP    eSI
4852 000029F3 6681C60020          <1>       ADD    SI, 2000h
4853 000029F8 6681C70020          <1>       ADD    DI, 2000h          ; POINT TO THE ODD FIELD
4854 000029FD 56                  <1>       PUSH   eSI
4855 000029FE 57                  <1>       PUSH   eDI                ; SAVE THE POINTERS
4856 000029FF 88D1                <1>       MOV    CL, DL             ; COUNT BACK
4857 00002A01 F3A4                <1>       REP    MOVSB              ; MOVE THE ODD FIELD
4858 00002A03 5F                  <1>       POP    eDI
4859 00002A04 5E                  <1>       POP    eSI                ; POINTERS BACK
4860 00002A05 C3                  <1>       RETn                      ; RETURN TO CALLER
4861                              <1>
4862                              <1> ;-----      CLEAR A SINGLE ROW
4863                              <1>
4864                              <1> _R18:
4865 00002A06 0FB6CA              <1>       MOVzx  ecx, DL                 ; NUMBER OF BYTES IN FIELD
4866 00002A09 57                  <1>       PUSH   eDI                ; SAVE POINTER
4867 00002A0A F3AA                <1>       REP    STOSB              ; STORE THE NEW VALUE
4868 00002A0C 5F                  <1>       POP    eDI                ; POINTER BACK
4869 00002A0D 6681C70020          <1>       ADD    DI, 2000h          ; POINT TO ODD FIELD
4870 00002A12 57                  <1>       PUSH   eDI
4871 00002A13 88D1                <1>       MOV    CL, DL
4872 00002A15 F3AA                <1>       REP    STOSB              ; FILL THE ODD FIELD
4873 00002A17 5F                  <1>       POP    eDI
4874 00002A18 C3                  <1>       RETn                      ; RETURN TO CALLER
4875                              <1>
4876                              <1> ; 04/07/2016
4877                              <1> ; 01/07/2016
4878                              <1> ; 30/06/2016 - TRDOS 386 (TRDOS v2.0)
4879                              <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
4880                              <1> ;-------------------------------------------------
4881                              <1> ; GRAPHICS WRITE
4882                              <1> ;  THIS ROUTINE WRITES THE ASCII CHARACTER TO THE CURRENT
4883                              <1> ;  POSITION ON THE SCREEN.
4884                              <1> ; ENTRY --
4885                              <1> ;  AL = CHARACTER TO WRITE
4886                              <1> ;  BL = COLOR ATTRIBUTE TO BE USED FOR FOREGROUND COLOR
4887                              <1> ;     IF BIT 7 IS SET, THE CHAR IS XOR'D INTO THE REGEN BUFFER
4888                              <1> ;     (0 IS USED FOR THE BACKGROUND COLOR)
4889                              <1> ;  CX = NUMBER OF CHARS TO WRITE
4890                              <1> ;  DS = DATA SEGMENT
4891                              <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:
00002A19 25FF000000     <1>       and   eax, 0FFh           ; ZERO TO HIGH OF CODE POINT
00002A1E 50             <1>       PUSH  eAX                 ; SAVE CODE POINT VALUE
                        <1>
                        <1> ;-----      DETERMINE POSITION IN REGEN BUFFER TO PUT CODE POINTS
                        <1>
00002A1F E84D010000     <1>       CALL  S26                 ; FIND LOCATION IN REGEN BUFFER
00002A24 89C7           <1>       MOV   eDI, eAX            ; REGEN POINTER IN DI
                        <1>
                        <1> ;-----      DETERMINE REGION TO GET CODE POINTS FROM
                        <1>
00002A26 58             <1>       POP   eAX                 ; RECOVER CODE POINT
                        <1>
00002A27 BE[24580100]   <1>       MOV   eSI, CRT_CHAR_GEN   ; OFFSET OF IMAGES
                        <1>
                        <1> ;-----      DETERMINE GRAPHICS MODE IN OPERATION
                        <1>                                 ; DETERMINE_MODE
00002A2C 66C1E003       <1>       SAL   AX, 3               ; MULTIPLY CODE POINT VALUE BY 8
00002A30 01C6           <1>       ADD   eSI, eAX            ; SI HAS OFFSET OF DESIRED CODES
                        <1>
00002A32 803D[BA6F0000]06 <1>     CMP   byte [CRT_MODE], 6
00002A39 7231           <1>       JC    short S6            ; TEST FOR MEDIUM RESOLUTION MODE
                        <1>
                        <1> ;-----      HIGH RESOLUTION MODE
                        <1>
00002A3B 81C700800B00   <1>       add   edi, 0B8000h
                        <1> S1:                             ; HIGH_CHAR
00002A41 57             <1>       PUSH  eDI                 ; SAVE REGEN POINTER
00002A42 56             <1>       PUSH  eSI                 ; SAVE CODE POINTER
00002A43 B604           <1>       MOV   DH, 4               ; NUMBER OF TIMES THROUGH LOOP
                        <1> S2:
00002A45 AC             <1>       LODSB                     ; GET BYTE FROM CODE POINTS
00002A46 F6C380         <1>       TEST  BL, 80H                   ; SHOULD WE USE THE FUNCTION
00002A49 7515           <1>       JNZ   short S5            ; TO PUT CHAR IN
00002A4B AA             <1>       STOSB                     ; STORE IN REGEN BUFFER
00002A4C AC             <1>       LODSB
                        <1> S4:
00002A4D 8887FF1F0000   <1>       MOV   [eDI+2000H-1], AL ; STORE IN SECOND HALF
00002A53 83C74F         <1>       ADD   eDI, 79                   ; MOVE TO NEXT ROW IN REGEN
00002A56 FECE           <1>       DEC   DH                  ; DONE WITH LOOP
00002A58 75EB           <1>       JNZ   short S2
00002A5A 5E             <1>       POP   eSI
00002A5B 5F             <1>       POP   eDI                 ; RECOVER REGEN POINTER
00002A5C 47             <1>       INC   eDI                 ; POINT TO NEXT CHAR POSITION
00002A5D E2E2           <1>       LOOP  S1                  ; MORE CHARS TO WRITE
00002A5F C3             <1>       retn
                        <1>
                        <1> S5:
00002A60 3207           <1>       XOR   AL, [eDI]           ; EXCLUSIVE OR WITH CURRENT
00002A62 AA             <1>       STOSB                     ; STORE THE CODE POINT
00002A63 AC             <1>       LODSB                     ; AGAIN FOR ODD FIELD
00002A64 3287FF1F0000   <1>       XOR   AL, [eDI+2000H-1]
00002A6A EBE1           <1>       JMP   short S4            ; BACK TO MAINSTREAM
                        <1>
                        <1> ;-----      MEDIUM RESOLUTION WRITE
                        <1> S6:                             ; MED_RES_WRITE
00002A6C 88DA           <1>       MOV   DL, BL              ; SAVE HIGH COLOR BIT
00002A6E 66D1E7         <1>       SAL   DI, 1               ; OFFSET*2 SINCE 2 BYTES/CHAR
                        <1>                                 ; EXPAND BL TO FULL WORD OF COLOR
00002A71 80E303         <1>       AND   BL, 3               ; ISOLATE THE COLOR BITS ( LOW 2 BITS )
00002A74 B055           <1>       MOV   AL, 055H            ; GET BIT CONVERSION MULTIPLIER
00002A76 F6E3           <1>       MUL   BL                  ; EXPAND 2 COLOR BITS TO 4 REPLICATIONS
00002A78 88C3           <1>       MOV   BL, AL              ; PLACE BACK IN WORK REGISTER
00002A7A 88C7           <1>       MOV   BH, AL              ; EXPAND TO 8 REPLICATIONS OF COLOR BITS
00002A7C 81C700800B00   <1>       add   edi, 0B8000h
                        <1> S7:                                   ; MED_CHAR
00002A82 57             <1>       PUSH  eDI                 ; SAVE REGEN POINTER
00002A83 56             <1>       PUSH  eSI                 ; SAVE THE CODE POINTER
00002A84 B604           <1>       MOV   DH, 4               ; NUMBER OF LOOPS
                        <1> S8:
00002A86 AC             <1>       LODSB                     ; GET CODE POINT
00002A87 E8B3000000     <1>       CALL  S21                 ; DOUBLE UP ALL THE BITS
00002A8C 6621D8         <1>       AND   AX, BX              ; CONVERT TO FOREGROUND COLOR ( 0 BACK )
00002A8F 86E0           <1>       XCHG  AH, AL              ; SWAP HIGH/LOW BYTES FOR WORD MOVE
00002A91 F6C280         <1>       TEST  DL, 80H                   ; IS THIS XOR FUNCTION
00002A94 7403           <1>       JZ    short S9            ; NO, STORE IT IN AS IS
00002A96 663307         <1>       XOR   AX, [eDI]           ; DO FUNCTION WITH LOW/HIGH
                        <1> S9:
00002A99 668907         <1>       MOV   [eDI], AX           ; STORE FIRST BYTE HIGH, SECOND LOW
00002A9C AC             <1>       LODSB                     ; GET CODE POINT
00002A9D E89D000000     <1>       CALL  S21
00002AA2 6621D8         <1>       AND   AX, BX              ; CONVERT TO COLOR
00002AA5 86E0           <1>       XCHG  AH, AL              ; SWAP HIGH/LOW BYTES FOR WORD MOVE
00002AA7 F6C280         <1>       TEST  DL, 80H                   ; AGAIN, IS THIS XOR FUNCTION
00002AAA 7407           <1>       JZ    short _S10          ; NO, JUST STORE THE VALUES
00002AAC 66338700200000 <1>       XOR   AX, [eDI+2000H]           ; FUNCTION WITH FIRST HALF LOW
```

```
4997                                    <1> _S10:
4998 00002AB3 66898700200000            <1>       MOV   [eDI+2000H], AX           ; STORE SECOND PORTION HIGH
4999 00002ABA 6683C750                  <1>       ADD   DI, 80              ; POINT TO NEXT LOCATION
5000 00002ABE FECE                      <1>       DEC   DH
5001 00002AC0 75C4                      <1>       JNZ   short S8            ; KEEP GOING
5002 00002AC2 5E                        <1>       POP   eSI                 ; RECOVER CODE POINTER
5003 00002AC3 5F                        <1>       POP   eDI                 ; RECOVER REGEN POINTER
5004 00002AC4 47                        <1>       INC   eDI                 ; POINT TO NEXT CHAR POSITION
5005 00002AC5 47                        <1>       INC   eDI
5006 00002AC6 E2BA                      <1>       LOOP  S7                  ; MORE TO WRITE
5007 00002AC8 C3                        <1>       retn
5008                                    <1>
5009                                    <1> ; 04/07/2016
5010                                    <1> ; 01/07/2016
5011                                    <1> ; 30/06/2016 - TRDOS 386 (TRDOS v2.0)
5012                                    <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
5013                                    <1> ;----------------------------------------
5014                                    <1> ; GRAPHICS READ
5015                                    <1> ;----------------------------------------
5016                                    <1> GRAPHICS_READ:
5017 00002AC9 E8A3000000                <1>       CALL  S26                 ; CONVERTED TO OFFSET IN REGEN
5018 00002ACE 89C6                      <1>       MOV   eSI, eAX            ; SAVE IN SI
5019 00002AD0 81C600800B00              <1>       add   esi, 0B8000h        ; 01/07/2016
5020 00002AD6 83EC08                    <1>       SUB   eSP, 8              ; ALLOCATE SPACE FOR THE READ CODE POINT
5021 00002AD9 89E5                      <1>       MOV   eBP, eSP            ; POINTER TO SAVE AREA
5022                                    <1>
5023                                    <1> ;-----      DETERMINE GRAPHICS MODES
5024 00002ADB B604                      <1>       mov   dh, 4               ; number of passes ; 01/07/2016
5025 00002ADD 803D[BA6F0000]06          <1>       CMP   byte [CRT_MODE], 6
5026 00002AE4 7219                      <1>       JC    short S12           ; MEDIUM RESOLUTION
5027                                    <1>
5028                                    <1> ;-----      HIGH RESOLUTION READ
5029                                    <1> ;-----      GET VALUES FROM REGEN BUFFER AND CONVERT TO CODE POINT
5030                                    <1>       ;MOV  DH,4                ; NUMBER OF PASSES
5031                                    <1> S11:
5032 00002AE6 8A06                      <1>       MOV   AL, [eSI]           ; GET FIRST BYTE
5033 00002AE8 884500                    <1>       MOV   [eBP], AL           ; SAVE IN STORAGE AREA
5034 00002AEB 45                        <1>       INC   eBP                 ; NEXT LOCATION
5035 00002AEC 8A8600200000              <1>       MOV   AL, [eSI+2000H]           ; GET LOWER REGION BYTE
5036 00002AF2 884500                    <1>       MOV   [eBP], AL           ; ADJUST AND STORE
5037 00002AF5 45                        <1>       INC   eBP
5038 00002AF6 83C650                    <1>       ADD   eSI, 80                   ; POINTER INTO REGEN
5039 00002AF9 FECE                      <1>       DEC   DH                  ; LOOP CONTROL
5040 00002AFB 75E9                      <1>       JNZ   short S11           ; DO IT SOME MORE
5041 00002AFD EB1D                      <1>       JMP   SHORT S14           ; GO MATCH THE SAVED CODE POINTS
5042                                    <1>
5043                                    <1> ;-----      MEDIUM RESOLUTION READ
5044                                    <1> S12:
5045 00002AFF 66D1E6                    <1>       SAL   SI, 1               ; OFFSET*2 SINCE 2 BYTES/CHAR
5046                                    <1>       ;MOV  DH, 4               ; NUMBER OF PASSES
5047                                    <1> S13:
5048 00002B02 E84D000000                <1>       CALL  S23                 ; GET BYTES FROM REGEN INTO SINGLE SAVE
5049 00002B07 81C6FE1F0000              <1>       ADD   eSI, 2000H-2        ; GO TO LOWER REGION
5050 00002B0D E842000000                <1>       CALL  S23                 ; GET THIS PAIR INTO SAVE
5051 00002B12 81EEB21F0000              <1>       SUB   eSI, 2000H-80+2           ; ADJUST POINTER BACK INTO UPPER
5052 00002B18 FECE                      <1>       DEC   DH
5053 00002B1A 75E6                      <1>       JNZ   short S13           ; KEEP GOING UNTIL ALL 8 DONE
5054                                    <1>
5055                                    <1> ;-----      SAVE AREA HAS CHARACTER IN IT, MATCH IT
5056                                    <1> S14:                              ; FIND_CHAR
5057 00002B1C BF[24580100]              <1>       MOV   eDI, CRT_CHAR_GEN   ; ESTABLISH ADDRESSING
5058 00002B21 83ED08                    <1>       SUB   eBP, 8              ; ADJUST POINTER TO START OF SAVE AREA
5059 00002B24 89EE                      <1>       MOV   eSI, eBP
5060                                    <1> S15:
5061 00002B26 66B80001                  <1>       mov   ax, 256                   ; NUMBER TO TEST AGAINST
5062                                    <1> S16:
5063 00002B2A 56                        <1>       PUSH  eSI                 ; SAVE SAVE AREA POINTER
5064 00002B2B 57                        <1>       PUSH  eDI                 ; SAVE CODE POINTER
5065                                    <1>       ;MOV  eCX, 4              ; NUMBER OF WORDS TO MATCH
5066                                    <1>       ;REPE CMPSW               ; COMPARE THE 8 BYTES AS WORDS
5067 00002B2C A7                        <1>       cmpsd                     ; compare first 4 bytes
5068 00002B2D 7501                      <1>       jne   short S17           ;
5069 00002B2F A7                        <1>       cmpsd                     ; compare last 4 bytes
5070                                    <1> S17:
5071 00002B30 5F                        <1>       POP   eDI                 ; RECOVER THE POINTERS
5072 00002B31 5E                        <1>       POP   eSI
5073                                    <1>       ;JZ   short S18           ; IF ZERO FLAG SET, THEN MATCH OCCURRED
5074 00002B32 7407                      <1>       je    short S18
5075                                    <1>       ;                         ; NO MATCH, MOVE ON TO NEXT
5076 00002B34 83C708                    <1>       ADD   eDI, 8              ; NEXT CODE POINT
5077 00002B37 6648                      <1>       dec   ax                  ; LOOP CONTROL
5078 00002B39 75EF                      <1>       JNZ   short S16           ; DO ALL OF THEM
5079                                    <1>
5080                                    <1> ;-----      CHARACTER IS FOUND ( AL=0 IF NOT FOUND )
5081                                    <1> S18:
5082 00002B3B 83C408                    <1>       ADD   eSP, 8              ; READJUST THE STACK, THROW AWAY SAVE
5083 00002B3E C3                        <1>       retn                      ; ALL DONE
5084                                    <1>
5085                                    <1> ; 30/06/2016 - TRDOS 386 (TRDOS v2.0)
5086                                    <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
5087                                    <1> ;----------------------------------------------
5088                                    <1> ; EXPAND BYTE
5089                                    <1> ;  THIS ROUTINE TAKES THE BYTE IN AL AND DOUBLES ALL
5090                                    <1> ;  OF THE BITS, TURNING THE 8 BITS INTO 16 BITS.
5091                                    <1> ;  THE RESULT IS LEFT IN AX
5092                                    <1> ;----------------------------------------------
5093                                    <1> S21:
5094 00002B3F 6651                      <1>       PUSH  CX                  ; SAVE REGISTER
5095                                    <1>       ;MOV  CX, 8               ; SHIFT COUNT REGISTER FOR ONE BYTE
5096 00002B41 B108                      <1>       mov   cl, 8
5097                                    <1> S22:
5098 00002B43 D0C8                      <1>       ROR   AL,1                ; SHIFT BITS, LOW BIT INTO CARRY FLAG
5099 00002B45 66D1DD                    <1>       RCR   BP,1                ; MOVE CARRY FLAG (LOW BIT INTO RESULTS
5100 00002B48 66D1FD                    <1>       SAR   BP,1                ; SIGN EXTEND HIGH BIT (DOUBLE IT)
5101                                    <1>       ;LOOP S22                 ; REPEAT FOR ALL 8 BITS
```

```
5102 00002B4B FEC9              <1>       dec    cl
5103 00002B4D 75F4              <1>       jnz    short S22
5104 00002B4F 6695              <1>       XCHG   AX, BP              ; MOVE RESULTS TO PARAMETER REGISTER
5105 00002B51 6659              <1>       POP    CX                  ; RECOVER REGISTER
5106 00002B53 C3                <1>       RETn                       ; ALL DONE
5107                            <1>
5108                            <1> ; 01/07/2016 - TRDOS 386 (TRDOS v2.0)
5109                            <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
5110                            <1> ;--------------------------------------------------
5111                            <1> ; MED_READ_BYTE
5112                            <1> ; THIS ROUTINE WILL TAKE 2 BYTES FROM THE REGEN BUFFER,
5113                            <1> ;  COMPARE AGAINST THE CURRENT FOREGROUND COLOR, AND PLACE
5114                            <1> ;  THE CORRESPONDING ON/OFF BIT PATTERN INTO THE CURRENT
5115                            <1> ;  POSITION IN THE SAVE AREA
5116                            <1> ; ENTRY --
5117                            <1> ;  SI,DS = POINTER TO REGEN AREA OF INTEREST
5118                            <1> ;  BX = EXPANDED FOREGROUND COLOR
5119                            <1> ;  BP = POINTER TO SAVE AREA
5120                            <1> ; EXIT --
5121                            <1> ;  SI AND BP ARE INCREMENTED
5122                            <1> ;--------------------------------------------------
5123                            <1> S23:
5124 00002B54 66AD              <1>       LODSW                      ; GET FIRST BYTE AND SECOND BYTES
5125 00002B56 86C4              <1>       XCHG   AL, AH              ; SWAP FOR COMPARE
5126 00002B58 66B900C0          <1>       MOV    CX, 0C000H          ; 2 BIT MASK TO TEST THE ENTRIES
5127 00002B5C B200              <1>       MOV    DL, 0               ; RESULT REGISTER
5128                            <1> S24:
5129 00002B5E 6685C8            <1>       TEST   AX, CX              ; IS THIS SECTION BACKCROUND?
5130 00002B61 7401              <1>         JZ     short S25               ; IF ZERO, IT IS BACKGROUND (CARRY=0)
5131 00002B63 F9                <1>       STC                        ; WASN'T, SO SET CARRY
5132                            <1> S25:
5133 00002B64 D0D2              <1>       RCL    DL, 1               ; MOVE THAT BIT INTO THE RESULT
5134 00002B66 66C1E902          <1>       SHR    CX, 2               ; MOVE THE MASK TO THE RIGHT BY 2 BITS
5135 00002B6A 73F2              <1>       JNC    short S24           ; DO IT AGAIN IF MASK DIDN'T FALL OUT
5136 00002B6C 885500            <1>       MOV    [eBP], DL           ; STORE RESULT IN SAVE AREA
5137 00002B6F 45                <1>       INC    eBP                 ; ADJUST POINTER
5138 00002B70 C3                <1>       RETn                       ; ALL DONE
5139                            <1>
5140                            <1> ; 30/06/2016 - TRDOS 386 (TRDOS v2.0)
5141                            <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
5142                            <1> ;-----------------------------------------
5143                            <1> ; V4_POSITION
5144                            <1> ;  THIS ROUTINE TAKES THE CURSOR POSITION CONTAINED IN
5145                            <1> ;  THE MEMORY LOCATION, AND CONVERTS IT INTO AN OFFSET
5146                            <1> ;  INTO THE REGEN BUFFER, ASSUMING ONE BYTE/CHAR.
5147                            <1> ;  FOR MEDIUM RESOLUTION GRAPHICS, THE NUMBER MUST
5148                            <1> ;  BE DOUBLED.
5149                            <1> ; ENTRY -- NO REGISTERS,MEMORY LOCATION @CURSOR_POSN IS USED
5150                            <1> ; EXIT--
5151                            <1> ;  AX CONTAINS OFFSET INTO REGEN BUFFER
5152                            <1> ;-----------------------------------------
5153                            <1> S26:
5154 00002B71 0FB705[AE810100]  <1>       movzx  eax, word [CURSOR_POSN]   ; GET CURRENT CURSOR
5155                            <1> GRAPH_POSN:
5156 00002B78 53                <1>       PUSH   eBX                 ; SAVE REGISTER
5157 00002B79 0FB6D8            <1>       movzx  ebx, al                   ; SAVE A COPY OF CURRENT CURSOR
5158 00002B7C A0[BC6F0000]      <1>       MOV    AL, [CRT_COLS]            ; GET BYTES PER COLUMN
5159 00002B81 F6E4              <1>       MUL    AH                  ; MULTIPLY BY ROWS
5160 00002B83 66C1E002          <1>       SHL    AX, 2               ; MULTIPLY * 4 SINCE 4 ROWS/BYTE
5161 00002B87 01D8              <1>       ADD    eAX, eBX            ; DETERMINE OFFSET
5162 00002B89 5B                <1>       POP    eBX                 ; RECOVER POINTER
5163 00002B8A C3                <1>       RETn                       ; ALL DONE
5164                            <1>
5165                            <1> ; 09/07/2016
5166                            <1> ; 01/07/2016 - TRDOS 386 (TRDOS v2.0)
5167                            <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
5168                            <1> ;-----------------------------------------------
5169                            <1> ; SET_COLOR
5170                            <1> ;     THIS ROUTINE WILL ESTABLISH THE BACKGROUND COLOR, THE OVERSCAN COLOR,
5171                            <1> ;     AND THE FOREGROUND COLOR SET FOR MEDIUM RESOLUTION GRAPHICS
5172                            <1> ; INPUT
5173                            <1> ;     (BH) HAS COLOR ID
5174                            <1> ;           IF BH=0, THE BACKGROUND COLOR VALUE IS SET
5175                            <1> ;                 FROM THE LOW BITS OF BL (0-31)
5176                            <1> ;           IF BH=1, THE PALETTE SELECTION IS MADE
5177                            <1> ;                 BASED ON THE LOW BIT OF BL:
5178                            <1> ;                       0 = GREEN, RED, YELLOW FOR COLORS 1,2,3
5179                            <1> ;                       1 = BLUE, CYAN, MAGENTA FOR COLORS 1,2,3
5180                            <1> ;     (BL) HAS THE COLOR VALUE TO BE USED
5181                            <1> ; OUTPUT
5182                            <1> ;     THE COLOR SELECTION IS UPDATED
5183                            <1> ;-----------------------------------------------
5184                            <1> SET_COLOR:
5185 00002B8B 803D[BA6F0000]07  <1>         cmp     byte [CRT_MODE], 7      ; 09/07/2016
5186 00002B92 0F87BFEFFFFF      <1>       ja     VIDEO_RETURN        ; nothing to do for VGA modes
5187                            <1>
5188                            <1>       ;MOV   DX, [ADDR_6845]           ; I/O PORT FOR PALETTE
5189                            <1>       ;mov   dx, 3D4h
5190                            <1>       ;ADD   DX,5                ; OVERSCAN PORT
5191 00002B98 66BAD903          <1>       mov    dx, 3D9h
5192 00002B9C A0[BD6F0000]      <1>       MOV    AL, [CRT_PALETTE]   ; GET THE CURRENT PALETTE VALUE
5193 00002BA1 08FF              <1>       OR     BH, BH              ; IS THIS COLOR 0?
5194 00002BA3 7512              <1>       JNZ    short M20           ; OUTPUT COLOR 1
5195                            <1>
5196                            <1> ;-----      HANDLE COLOR 0 BY SETTING THE BACKGROUND COLOR
5197                            <1>
5198 00002BA5 24E0              <1>       AND    AL, 0E0H            ; TURN OFF LOW 5 BITS OF CURRENT
5199 00002BA7 80E31F            <1>       AND    BL, 01FH            ; TURN OFF HIGH 3 BITS OF INPUT VALUE
5200 00002BAA 08D8              <1>       OR     AL, BL              ; PUT VALUE INTO REGISTER
5201                            <1> M19:                             ; OUTPUT THE PALETTE
5202 00002BAC EE                <1>       OUT    DX, AL              ; OUTPUT COLOR SELECTION TO 3D9 PORT
5203 00002BAD A2[BD6F0000]      <1>       MOV    [CRT_PALETTE], AL   ; SAVE THE COLOR VALUE
5204 00002BB2 E9A0EFFFFF        <1>       JMP    VIDEO_RETURN
5205                            <1>
5206                            <1> ;-----      HANDLE COLOR 1 BY SELECTING THE PALETTE TO BE USED
```

```
                                <1>
                                <1> M20:
00002BB7 24DF                   <1>         AND     AL, 0DFH            ; TURN OFF PALETTE SELECT BIT
00002BB9 D0EB                   <1>         SHR     BL, 1               ; TEST THE LOW ORDER BIT OF BL
00002BBB 73EF                   <1>         JNC     short M19           ; ALREADY DONE
00002BBD 0C20                   <1>         OR      AL, 20H                   ; TURN ON PALETTE SELECT BIT
00002BBF EBEB                   <1>         JMP     short M19           ; GO DO IT
                                <1>
                                <1> ; 09/07/2016
                                <1> ; 01/07/2016 - TRDOS 386 (TRDOS v2.0)
                                <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
                                <1> ;-----------------------------------------
                                <1> ; READ DOT -- WRITE DOT
                                <1> ; THESE ROUTINES WILL WRITE A DOT, OR READ THE
                                <1> ;  DOT AT THE INDICATED LOCATION
                                <1> ; ENTRY --
                                <1> ;   DX = ROW (0-199)      (THE ACTUAL VALUE DEPENDS ON THE MODE)
                                <1> ;   CX = COLUMN ( 0-639) ( THE VALUES ARE NOT RANGE CHECKED )
                                <1> ;   AL = DOT VALUE TO WRITE (1,2 OR 4 BITS DEPENDING ON MODE,
                                <1> ;     REQUIRED FOR WRITE DOT ONLY, RIGHT JUSTIFIED)
                                <1> ;     BIT 7 OF AL = 1 INDICATES XOR THE VALUE INTO THE LOCATION
                                <1> ;   DS = DATA SEGMENT
                                <1> ;   ES = REGEN SEGMENT
                                <1> ;
                                <1> ; EXIT
                                <1> ;     AL = DOT VALUE READ, RIGHT JUSTIFIED, READ ONLY
                                <1> ;-----------------------------------------------
                                <1>
                                <1> READ_DOT:
                                <1>         ; 09/07/2016
00002BC1 8A25[BA6F0000]         <1>         mov     ah,  [CRT_MODE]
00002BC7 80FC07                 <1>         cmp     ah, 7 ; 6!?
00002BCA 760A                   <1>         jna     short read_dot_cga
                                <1>
00002BCC E8D7030000             <1>         call    vga_read_pixel
                                <1>         ; al = pixel value
00002BD1 E986EFFFFF             <1>         jmp     _video_return
                                <1>
                                <1> read_dot_cga:
                                <1>         ;je     VIDEO_RETURN ; 7
00002BD6 80FC04                 <1>         cmp     ah, 4 ; graphics ?
00002BD9 0F8278EFFFFF           <1>         jb      VIDEO_RETURN ; no, text mode, nothing to do
                                <1>
00002BDF E855000000             <1>         CALL    R3                  ; DETERMINE BYTE POSITION OF DOT
00002BE4 8A06                   <1>         MOV     AL, [eSI]           ; GET THE BYTE
00002BE6 20E0                   <1>         AND     AL, AH              ; MASK OFF THE OTHER BITS IN THE BYTE
00002BE8 D2E0                   <1>         SHL     AL, CL              ; LEFT JUSTIFY THE VALUE
00002BEA 88F1                   <1>         MOV     CL, DH              ; GET NUMBER OF BITS IN RESULT
00002BEC D2C0                   <1>         ROL     AL, CL              ; RIGHT JUSTIFY THE RESULT
                                <1>         ;JMP    VIDEO_RETURN        ; RETURN FROM VIDEO I/O
00002BEE 0FB6C0                 <1>         movzx   eax, al
00002BF1 E966EFFFFF             <1>         jmp     _video_return
                                <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> WRITE_DOT:
                                <1>         ; 09/07/2016
00002BF6 8A25[BA6F0000]         <1>         mov     ah, [CRT_MODE]
00002BFC 80FC07                 <1>         cmp     ah, 7 ; 6!?
00002BFF 760A                   <1>         jna     short write_dot_cga
                                <1>
00002C01 E811030000             <1>         call    vga_write_pixel
00002C06 E94CEFFFFF             <1>         jmp     VIDEO_RETURN
                                <1>
                                <1> write_dot_cga:
                                <1>         ;je     VIDEO_RETURN ; 7
00002C0B 80FC04                 <1>         cmp     ah, 4 ; graphics ?
00002C0E 0F8243EFFFFF           <1>         jb      VIDEO_RETURN ; no, text mode, nothing to do
                                <1>
                                <1>         ;PUSH   AX                  ; SAVE DOT VALUE
00002C14 6650                   <1>         PUSH    AX                  ; TWICE
00002C16 E81E000000             <1>         CALL    R3                  ; DETERMINE BYTE POSITION OF THE DOT
00002C1B D2E8                   <1>         SHR     AL, CL              ; SHIFT TO SET UP THE BITS FOR OUTPUT
00002C1D 20E0                   <1>         AND     AL, AH              ; STRIP OFF THE OTHER BITS
00002C1F 8A0E                   <1>         MOV     CL, [eSI]           ; GET THE CURRENT BYTE
00002C21 665B                   <1>         POP     BX                  ; RECOVER XOR FLAG
00002C23 F6C380                 <1>         TEST    BL, 80H                   ; IS IT ON
00002C26 750D                   <1>         JNZ     short R2            ; YES, XOR THE DOT
00002C28 F6D4                   <1>         NOT     AH                  ; SET MASK TO REMOVE THE INDICATED BITS
00002C2A 20E1                   <1>         AND     CL, AH
00002C2C 08C8                   <1>         OR      AL, CL              ; OR IN THE NEW VALUE OF THOSE BITS
                                <1> R1:                                 ; FINISH_DOT
00002C2E 8806                   <1>         MOV     [eSI], AL           ; RESTORE THE BYTE IN MEMORY
                                <1>         ;POP    AX
00002C30 E922EFFFFF             <1>         JMP     VIDEO_RETURN        ; RETURN FROM VIDEO I/O
                                <1> R2:                                 ; XOR_DOT
00002C35 30C8                   <1>         XOR     AL, CL              ; EXCLUSIVE OR THE DOTS
00002C37 EBF5                   <1>         JMP     short R1            ; FINISH UP THE WRITING
                                <1>
                                <1> ; 01/07/2016 - TRDOS 386 (TRDOS v2.0)
                                <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
                                <1>
                                <1> ;-----------------------------------------------
                                <1> ; THIS SUBROUTINE DETERMINES THE REGEN BYTE LOCATION OF THE
                                <1> ; INDICATED ROW COLUMN VALUE IN GRAPHICS MODE.
                                <1> ; ENTRY --
                                <1> ;  DX = ROW VALUE (0-199)
                                <1> ;  CX = COLUMN VALUE (0-639)
                                <1> ; EXIT --
                                <1> ;  SI = OFFSET INTO REGEN BUFFER FOR BYTE OF INTEREST
                                <1> ;  AH = MASK TO STRIP OFF THE BITS OF INTEREST
                                <1> ;  CL = BITS TO SHIFT TO RIGHT JUSTIFY THE MASK IN AH
                                <1> ;  DH = # BITS IN RESULT
```

```
                        <1> ;  BX = MODIFIED
                        <1> ;-------------------------------------------------
                        <1> R3:
                        <1>
                        <1> ;-----      DETERMINE 1ST BYTE IN INDICATED ROW BY MULTIPLYING ROW VALUE BY 40
                        <1> ;-----       ( LOW BIT OF ROW DETERMINES EVEN/ODD, 80 BYTES/ROW )
                        <1>
00002C39 0FB7F0         <1>       movzx  esi, ax                  ; WILL SAVE AL AND AH DURING OPERATION
00002C3C B028           <1>       MOV    AL, 40
00002C3E F6E2           <1>       MUL    DL                 ; AX= ADDRESS OF START OF INDICATED ROW
00002C40 A808           <1>       TEST   AL, 08H            ; TEST FOR EVEN/ODD ROW CALCULATED
00002C42 7404           <1>       JZ     short R4           ; JUMP IF EVEN ROW
00002C44 6605D81F       <1>       ADD    AX, 2000H-40       ; OFFSET TO LOCATION OF ODD ROWS ADJUST
                        <1> R4:                             ; EVEN_ROW
00002C48 6696           <1>       XCHG   SI, AX             ; MOVE POINTER TO (SI) AND RECOVER (AX)
00002C4A 81C600800B00   <1>       add    esi, 0B8000h
00002C50 6689CA         <1>       MOV    DX, CX             ; COLUMN VALUE TO DX
                        <1>
                        <1> ;-----      DETERMINE GRAPHICS MODE CURRENTLY IN EFFECT
                        <1>
                        <1> ; SET UP THE REGISTERS ACCORDING TO THE MODE
                        <1> ; CH = MASK FOR LOW OF COLUMN ADDRESS ( 7/3 FOR HIGH/MED RES )
                        <1> ; CL = # OF ADDRESS BITS IN COLUMN VALUE ( 3/2 FOR H/M )
                        <1> ; BL = MASK TO SELECT BITS FROM POINTED BYTE ( 80H/C0H FOR H/M )
                        <1> ; BH = NUMBER OF VALID BITS IN POINTED BYTE ( 1/2 FOR H/M )
                        <1>
00002C53 66BBC002       <1>       MOV    BX, 2C0H
00002C57 66B90203       <1>       MOV    CX, 302H           ; SET PARMS FOR MED RES
00002C5B 803D[BA6F0000]06 <1>     CMP    byte [CRT_MODE], 6
00002C62 7208           <1>       JC     short R5           ; HANDLE IF MED RES
00002C64 66BB8001       <1>       MOV    BX, 180H
00002C68 66B90307       <1>       MOV    CX, 703H           ; SET PARMS FOR HIGH RES
                        <1>
                        <1> ;-----      DETERMINE BIT OFFSET IN BYTE FROM COLUMN MASK
                        <1> R5:
00002C6C 20D5           <1>       AND    CH, DL             ; ADDRESS OF PEL WITHIN BYTE TO CH
                        <1>
                        <1> ;-----      DETERMINE BYTE OFFSET FOR THIS LOCATION IN COLUMN
                        <1>
00002C6E 66D3EA         <1>       SHR    DX, CL             ; SHIFT BY CORRECT AMOUNT
00002C71 6601D6         <1>       ADD    SI, DX             ; INCREMENT THE POINTER
00002C74 88FE           <1>       MOV    DH, BH             ; GET THE # OF BITS IN RESULT TO DH
                        <1>
                        <1> ;-----      MULTIPLY BH (VALID BITS IN BYTE) BY CH (BIT OFFSET)
                        <1>
00002C76 28C9           <1>       SUB    CL, CL             ; ZERO INTO STORAGE LOCATION
                        <1> R6:
00002C78 D0C8           <1>       ROR    AL, 1              ; LEFT JUSTIFY VALUE IN AL (FOR WRITE)
00002C7A 00E9           <1>       ADD    CL, CH             ; ADD IN THE BIT OFFSET VALUE
00002C7C FECF           <1>       DEC    BH                 ; LOOP CONTROL
00002C7E 75F8           <1>       JNZ    short R6           ; ON EXIT, CL HAS COUNT TO RESTORE BITS
00002C80 88DC           <1>       MOV    AH, BL             ;  GET MASK TO AH
00002C82 D2EC           <1>       SHR    AH, CL             ;  MOVE THE MASK TO CORRECT LOCATION
00002C84 C3             <1>       RETn                      ;  RETURN WITH EVERYTHING SET UP
                        <1>
                        <1> load_dac_palette:
                        <1>       ; 29/07/2016
                        <1>       ; 23/07/2016
                        <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', 'load_dac_palette'
                        <1>       ;
                        <1>       ; Oracle VirtualBox 5.0.24 VGABios Source Code
                        <1>       ; ('vgabios.c', 'vgatables.h', 'vgafonts.h', 'vgarom.asm')
                        <1>       ;
                        <1>       ; INPUT -> AH = DAC selection number (3, 2 or 1)
                        <1>       ; OUTPUT -> ECX = 0, AX = 0
                        <1>       ; (Modifed registers: EAX, ECX, EDX, ESI)
                        <1>       ;
00002C85 66BAC803       <1>       mov    dx, 3C8h  ; VGAREG_DAC_WRITE_ADDRESS
00002C89 28C0           <1>       sub    al, al ; 0
00002C8B EE             <1>       out    dx, al ; 0 ; color index, always 0 at the beginning
00002C8C 6642           <1>       inc    dx   ; 3C9h ; VGAREG_DAC_DATA
00002C8E B900010000     <1>       mov    ecx, 256   ; always 256*3 values
                        <1>       ;push  esi
00002C93 88E0           <1>       mov    al, ah
00002C95 B43F           <1>       mov    ah, 3Fh      ; 3Fh except DAC selection number 3
00002C97 3C02           <1>       cmp    al, 2
00002C99 7414           <1>       je     short l_dac_p_2
00002C9B 7719           <1>       ja     short l_dac_p_3
00002C9D 20C0           <1>       and    al, al
00002C9F 7507           <1>       jnz    short l_dac_p_1
                        <1> l_dac_p_0:
00002CA1 BE[E4520100]   <1>       mov    esi, palette0
00002CA6 EB15           <1>       jmp    short l_dac_p_4
                        <1> l_dac_p_1:
00002CA8 BE[A4530100]   <1>       mov    esi, palette1
00002CAD EB0E           <1>       jmp    short l_dac_p_4
                        <1> l_dac_p_2:
00002CAF BE[64540100]   <1>       mov    esi, palette2
00002CB4 EB07           <1>       jmp    short l_dac_p_4
                        <1> l_dac_p_3:
00002CB6 B4FF           <1>       mov    ah, 0FFh ; dac registers
00002CB8 BE[24550100]   <1>       mov    esi, palette3
                        <1> l_dac_p_4:
00002CBD AC             <1>       lodsb
00002CBE EE             <1>       out    dx, al  ; Red
00002CBF AC             <1>       lodsb
00002CC0 EE             <1>       out    dx, al ; Green
00002CC1 AC             <1>       lodsb
00002CC2 EE             <1>       out    dx, al ; Blue
00002CC3 20E4           <1>       and    ah, ah
```

```
5417 00002CC5 7405              <1>       jz    short l_dac_p_5
5418 00002CC7 FECC              <1>       dec   ah
5419 00002CC9 E2F2              <1>       loop  l_dac_p_4
5420                            <1>       ;pop  esi
5421 00002CCB C3                <1>       retn
5422                            <1> l_dac_p_5:
5423                            <1>       ; 29/07/2016
5424 00002CCC FEC9              <1>       dec   cl
5425 00002CCE 7407              <1>       jz    short l_dac_p_7
5426                            <1>       ;
5427 00002CD0 28C0              <1>       sub   al, al ; 0
5428                            <1> l_dac_p_6:
5429 00002CD2 EE                <1>       out   dx, al ; outb(VGAREG_DAC_DATA,0);
5430 00002CD3 EE                <1>       out   dx, al
5431 00002CD4 EE                <1>       out   dx, al
5432 00002CD5 E2FB              <1>       loop  l_dac_p_6
5433                            <1> l_dac_p_7:
5434                            <1>       ;pop  esi
5435 00002CD7 C3                <1>       retn
5436                            <1>
5437                            <1> gray_scale_summing:
5438                            <1>       ; 03/07/2016 (TRDOS 386 = TRDOS v2.0)
5439                            <1>       ; (set_mode_vga)
5440                            <1>       ; derived from 'Plex86/Bochs VGABios' source code
5441                            <1>       ; vgabios-0.7a (2011)
5442                            <1>       ; by the LGPL VGABios developers Team (2001-2008)
5443                            <1>       ; 'vgabios.c', 'biosfn_perform_gray_scale_summing'
5444                            <1>       ;
5445                            <1>       ; Oracle VirtualBox 5.0.24 VGABios Source Code
5446                            <1>       ; ('vgabios.c', 'vgatables.h', 'vgafonts.h', 'vgarom.asm')
5447                            <1>       ;
5448                            <1>
5449                            <1>       ; INPUT -> EBX = Start address (color index <= 255)
5450                            <1>       ;         ECX = Count (<= 256)
5451                            <1>       ; OUTPUT -> (E)CX = 0
5452                            <1>       ; (Modifed registers: EAX, ECX, EDX, EBX)
5453                            <1>
5454 00002CD8 66BADA03          <1>       mov   dx, 3DAh ; VGAREG_ACTL_RESET
5455 00002CDC EC                <1>       in    al, dx
5456 00002CDD 30C0              <1>       xor   al, al ; 0
5457 00002CDF 66BAC003          <1>       mov   dx, 3C0h ; VGAREG_ACTL_ADDRESS
5458 00002CE3 EE                <1>       out   dx, al ; clear bit 5
5459                            <1>                    ; (while loading palette registers)
5460                            <1>       ; set read address and switch to read mode
5461                            <1> g_s_s_1:
5462 00002CE4 66BAC703          <1>       mov   dx, 3C7h ; VGAREG_DAC_READ_ADDRESS
5463 00002CE8 88D8              <1>       mov   al, bl
5464 00002CEA EE                <1>       out   dx, al
5465                            <1>       ; get 6-bit wide RGB data values
5466                            <1>       ; intensity = (0.3*Red)+(0.59*Green)+(0.11*Blue)
5467                            <1>       ; i = ( ( 77*r + 151*g + 28*b ) + 0x80 ) >> 8;
5468 00002CEB 66BAC903          <1>       mov   dx, 3C9h ; VGAREG_DAC_DATA
5469 00002CEF EC                <1>       in    al, dx ; red
5470 00002CF0 B44D              <1>       mov   ah, 77 ; 0.3* Red
5471 00002CF2 F6E4              <1>       mul   ah
5472 00002CF4 6650              <1>       push  ax
5473 00002CF6 EC                <1>       in    al, dx ; green
5474 00002CF7 B497              <1>       mov   ah, 151  ; 0.59 * Green
5475 00002CF9 F6E4              <1>       mul   ah
5476 00002CFB 6650              <1>       push  ax
5477 00002CFD EC                <1>       in    al, dx ; blue
5478 00002CFE B41C              <1>       mov   ah, 28 ; 0.11 * Blue
5479 00002D00 F6E4              <1>       mul   ah
5480 00002D02 665A              <1>       pop   dx
5481 00002D04 6601D0            <1>       add   ax, dx
5482 00002D07 665A              <1>       pop   dx
5483 00002D09 6601D0            <1>       add   ax, dx
5484 00002D0C 66058000          <1>       add   ax, 80h
5485 00002D10 B03F              <1>       mov   al, 3Fh
5486 00002D12 38C4              <1>       cmp   ah, al ; if(i>0x3f)i=0x3f
5487 00002D14 7602              <1>       jna   short g_s_s_2
5488 00002D16 88C4              <1>       mov   ah, al
5489                            <1> g_s_s_2:
5490 00002D18 66BAC803          <1>       mov   dx, 3C8h  ; VGAREG_DAC_WRITE_ADDRESS
5491 00002D1C 88D8              <1>       mov   al, bl ; color index
5492 00002D1E EE                <1>       out   dx, al
5493 00002D1F 88E0              <1>       mov   al, ah ; intensity
5494 00002D21 6642              <1>       inc   dx ; 3C9h ; VGAREG_DAC_DATA
5495 00002D23 EE                <1>       out   dx, al ; R (R=G=B)
5496 00002D24 88E0              <1>       mov   al, ah ; intensity
5497 00002D26 EE                <1>       out   dx, al ; G (R=G=B)
5498 00002D27 88E0              <1>       mov   al, ah ; intensity
5499 00002D29 EE                <1>       out   dx, al ; B (R=G=B)
5500 00002D2A 6649              <1>       dec   cx
5501 00002D2C 7404              <1>       jz    short g_s_s_3
5502 00002D2E FEC3              <1>       inc   bl    ; next color index value
5503 00002D30 EBB2              <1>       jmp   short g_s_s_1
5504                            <1> g_s_s_3:
5505 00002D32 66BADA03          <1>       mov   dx, 3DAh ; VGAREG_ACTL_RESET
5506 00002D36 EC                <1>       in    al, dx
5507 00002D37 B020              <1>       mov   al, 20h
5508 00002D39 66BAC003          <1>       mov   dx, 3C0h ; VGAREG_ACTL_ADDRESS
5509 00002D3D EE                <1>       out   dx, al ; 20h -> set bit 5
5510                            <1>                    ; (after loading palette regs)
5511 00002D3E C3                <1>       retn
5512                            <1>
5513                            <1> vga_write_char_attr:
5514                            <1> vga_write_char_only:
5515                            <1>       ; 08/07/2016 (TRDOS 386 = TRDOS v2.0)
5516                            <1>       ;
5517                            <1>       ; derived from 'Plex86/Bochs VGABios' source code
5518                            <1>       ; vgabios-0.7a (2011)
5519                            <1>       ; by the LGPL VGABios developers Team (2001-2008)
5520                            <1>       ; 'vgabios.c', 'biosfn_write_char_attr'
5521                            <1>       ; 'biosfn_write_char_only'
```

```
                                <1>
                                <1>       ; INPUT ->
                                <1>       ; [CRT_MODE] = current video mode (>7)
                                <1>       ; CX = Count of characters to write
                                <1>       ; AL = Character to write
                                <1>       ; BL = Color of character
                                <1>       ; OUTPUT ->
                                <1>       ; Regen buffer updated
                                <1>
00002D3F 8A25[BA6F0000]         <1>       mov    ah, [CRT_MODE]
00002D45 668B15[AE810100]       <1>       mov    dx, [CURSOR_POSN] ; cursor pos for page 0
                                <1>
00002D4C BE[D66F0000]           <1>       mov    esi, vga_modes
00002D51 89F7                   <1>       mov    edi, esi
00002D53 83C710                 <1>       add    edi, vga_mode_count
                                <1> vga_wca_0:
00002D56 AC                     <1>       lodsb
00002D57 38E0                   <1>       cmp    al, ah ; [CRT_MODE]
00002D59 7405                   <1>       je     short vga_wca_2
00002D5B 39FE                   <1>       cmp    esi, edi
00002D5D 72F7                   <1>       jb     short vga_wca_0
                                <1> vga_wca_1:
00002D5F C3                     <1>       retn   ; nothing to do
                                <1> vga_wca_2:
00002D60 83C64F                 <1>       add    esi, vga_memmodel - (vga_modes + 1)
                                <1>       ; [ESI] = VGA memory model number (LINEAR8, PLANAR4, PLANAR1)
                                <1>
                                <1>       ; biosfn_write_char_attr (car,page,attr,count)
                                <1>       ; AL = car, page = 0, BL = attr, CX = count
00002D63 803E04                 <1>       cmp    byte [esi], PLANAR4
00002D66 741D                   <1>       je     short vga_wca_planar
00002D68 803E03                 <1>       cmp    byte [esi], PLANAR1
00002D6B 7418                   <1>       je     short vga_wca_planar
                                <1> vga_wca_linear8:
                                <1>       ; while((count-->0) && (xcurs<nbcols))
                                <1>       ; CX = count
00002D6D 6621C9                 <1>       and    cx, cx
00002D70 74ED                   <1>       jz     short vga_wca_1
00002D72 3A15[BC6F0000]         <1>       cmp    dl, [CRT_COLS]
00002D78 73E5                   <1>       jnb    short vga_wca_1
                                <1>       ; write_gfx_char_lin(car,attr,xcurs,ycurs,nbcols);
                                <1>       ; AL = car, BL = attr, DL = xcurs, DH = ycurs,
                                <1>       ; [CRT_COLS] = nbcols
00002D7A E81E000000             <1>       call   write_gfx_char_lin
00002D7F 6649                   <1>       dec    cx ; count
00002D81 FEC2                   <1>       inc    dl ; xcurs
00002D83 EBE8                   <1>       jmp    short vga_wca_linear8
                                <1> vga_wca_planar:
                                <1>       ; while((count-->0) && (xcurs<nbcols))
                                <1>       ; CX = count
00002D85 6621C9                 <1>       and    cx, cx
00002D88 74D5                   <1>       jz     short vga_wca_1
00002D8A 3A15[BC6F0000]         <1>       cmp    dl, [CRT_COLS]
00002D90 73CD                   <1>       jnb    short vga_wca_1
                                <1>       ; write_gfx_char_pl4(car,attr,xcurs,ycurs,nbcols,cheight);
                                <1>       ; AL = car, BL = attr, DL = xcurs, DH = ycurs,
                                <1>       ; [CRT_COLS] = nbcols, [CHAR_HEIGHT] = cheight
00002D92 E8A9000000             <1>       call   write_gfx_char_pl4
00002D97 6649                   <1>       dec    cx ; count
00002D99 FEC2                   <1>       inc    dl ; xcurs
00002D9B EBE8                   <1>       jmp    short vga_wca_planar
                                <1>
                                <1> write_gfx_char_lin:
                                <1>       ; 08/01/2021
                                <1>       ; 05/01/2021 (TRDOS 386 v2.0.3)
                                <1>       ; 08/08/2016
                                <1>       ; 31/07/2016
                                <1>       ; 08/07/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>       ;
                                <1>       ; derived from 'Plex86/Bochs VGABios' source code
                                <1>       ; vgabios-0.7a (2011)
                                <1>       ; by the LGPL VGABios developers Team (2001-2008)
                                <1>       ; 'vgabios.c', 'write_gfx_char_lin'
                                <1>
                                <1>       ; write_gfx_char_lin(car,attr,xcurs,ycurs,nbcols)
                                <1>       ; INPUT ->
                                <1>       ; AL = car, BL = attr, DL = xcurs, DH = ycurs,
                                <1>       ; [CRT_COLS] = nbcols
                                <1>       ; OUTPUT ->
                                <1>       ; Regen buffer updated
                                <1>
00002D9D 51                     <1>       push   ecx
00002D9E 53                     <1>       push   ebx
00002D9F 52                     <1>       push   edx
00002DA0 50                     <1>       push   eax
                                <1>       ; addr=xcurs*8+ycurs*nbcols*64;
                                <1>       ; 08/08/2016
00002DA1 0FB6F0                 <1>       movzx  esi, al ; car
                                <1>       ; 08/01/2021
                                <1>       ;movzx eax, dh ; ycurs
                                <1>       ;mov   ah, [CRT_COLS] ; nbcols
                                <1>       ;mul   ah
00002DA4 A0[BC6F0000]           <1>       mov    al, [CRT_COLS]
00002DA9 F6E6                   <1>       mul    dh
                                <1>       ;shl   ax, 6 ; * 64
00002DAB 66C1E003               <1>       shl    ax, 3 ; 8 * ycurs * [CRT_COLS]
                                <1>       ;sub   dh, dh
                                <1>       ;shl   dx, 3 ; xcurs * 8
                                <1>       ;movzx edi, dx
00002DAF BF00000A00             <1>       mov    edi, 0A0000h
00002DB4 30F6                   <1>       xor    dh, dh
00002DB6 6689D7                 <1>       mov    di, dx
                                <1>       ;movzx edi, dl
00002DB9 66C1E703               <1>       shl    di, 3 ; xcurs * 8
                                <1>       ;xor   dh, dh
```

```
5627 00002DBD 8A15[BE6F0000]      <1>       mov    dl, [CHAR_HEIGHT]
5628 00002DC3 66F7E2              <1>       mul    dx
5629                              <1>       ; eax = ycurs*nbcols*8*[CHAR_HEIGHT]
5630                              <1>       ;add   edi, eax ; addr
5631                              <1>       ;add   edi, 0A0000h
5632 00002DC6 6601C7              <1>       add    di, ax
5633                              <1>       ;shl   si, 3 ; car * 8
5634 00002DC9 30E4                <1>       xor    ah, ah
5635 00002DCB A0[BE6F0000]        <1>       mov    al, [CHAR_HEIGHT]
5636 00002DD0 66F7E6              <1>       mul    si
5637 00002DD3 6689C6              <1>       mov    si, ax
5638                              <1>       ;; esi = src = car * 8
5639                              <1>       ; esi = src = car * [CHAR_HEIGHT]
5640                              <1>       ; i = 0
5641                              <1>       ;add   esi, vgafont8 ; fdata [src+i]
5642                              <1>       ; 08/08/2016
5643 00002DD6 A1[3A8E0100]        <1>       mov    eax, [VGA_INT43H]
5644 00002DDB 09C0                <1>       or     eax, eax ; 0 ?
5645 00002DDD 743F                <1>       jz     short wfxl_7 ; yes, default font
5646                              <1>       ;cmp   eax, vgafont16
5647                              <1>         ;je  short wgfxl_0
5648                              <1>       ;cmp   eax, vgafont14
5649                              <1>       ;je    short wgfxl_0
5650                              <1>       ;cmp   eax, vgafont8
5651                              <1>       ;je    short wgfxl_0
5652                              <1>       ;; 05/01/2021 (TRDOS 386 v2.0.3)
5653                              <1>       ;; user font
5654                              <1>       ;mov   eax, VGAFONTUSR ; 8x16 or 8x8 or 8x14 font
5655                              <1>       ;                      ; (256 characters)
5656                              <1> wgfxl_0:
5657 00002DDF 01C6                <1>       add    esi, eax
5658                              <1> wgfxl_1:
5659 00002DE1 28FF                <1>       sub    bh, bh ; i = 0
5660                              <1> wgfxl_2:
5661                              <1>       ; for(i=0;i<8;i++)
5662 00002DE3 57                  <1>       push   edi ; addr
5663 00002DE4 0FB605[BC6F0000]    <1>       movzx  eax, byte [CRT_COLS] ; nbcols
5664 00002DEB F6E7                <1>       mul    bh ; nbcols*i
5665 00002DED 66C1E003            <1>       shl    ax, 3 ; i*nbcols*8
5666                              <1>       ; dest=addr+i*nbcols*8;
5667 00002DF1 01C7                <1>       add    edi, eax ; dest + j ; j = 0
5668 00002DF3 B180                <1>       mov    cl, 80h ; mask = 0x80;
5669                              <1>       ; esi = fdata + src + i
5670                              <1>       ; for(j=0;j<8;j++)
5671 00002DF5 29D2                <1>       sub    edx, edx ; j = 0
5672                              <1> wgfxl_3:
5673 00002DF7 8A06                <1>       mov    al, [esi] ; al = fdata[src+i]
5674 00002DF9 20C8                <1>       and    al, cl ; if (fdata[src+i] & mask)
5675 00002DFB 7402                <1>       jz     short wgfxl_4  ; data = 0, zf = 1
5676 00002DFD 88D8                <1>       mov    al, bl ; data = attr;
5677                              <1> wgfxl_4:
5678                              <1>       ; write_byte(0xa000,dest+j,data);
5679 00002DFF AA                  <1>       stosb  ; dest + j (+ 0A0000h)
5680                              <1>       ;inc   dl ; j++
5681                              <1>       ;cmp   dl, 8
5682 00002E00 80FA07              <1>       cmp    dl, 7
5683 00002E03 720E                <1>       jb     short wgfxl_5
5684 00002E05 5F                  <1>       pop    edi
5685                              <1>       ; 08/08/2016
5686                              <1>       ;cmp   bh, 7
5687                              <1>       ;jnb   short wgfxl_6
5688 00002E06 FEC7                <1>       inc    bh ; i++
5689 00002E08 3A3D[BE6F0000]      <1>       cmp    bh, [CHAR_HEIGHT]
5690 00002E0E 7309                <1>       jnb    short wgfxl_6
5691 00002E10 46                  <1>       inc    esi
5692 00002E11 EBD0                <1>       jmp    short wgfxl_2
5693                              <1> wgfxl_5:
5694 00002E13 D0E9                <1>       shr    cl, 1 ; mask >>= 1;
5695 00002E15 FEC2                <1>       inc    dl ; j++
5696 00002E17 EBDE                <1>         jmp     short wgfxl_3
5697                              <1> wgfxl_6:
5698 00002E19 58                  <1>       pop    eax
5699 00002E1A 5A                  <1>       pop    edx
5700 00002E1B 5B                  <1>       pop    ebx
5701 00002E1C 59                  <1>       pop    ecx
5702 00002E1D C3                  <1>       retn
5703                              <1> wfxl_7:
5704                              <1>       ; 08/01/2021
5705                              <1>       ; 05/01/2021
5706                              <1>       ; Default font (8x8 or 8x14 or 8x16)
5707 00002E1E A0[BE6F0000]        <1>       mov    al, [CHAR_HEIGHT]
5708 00002E23 3C08                <1>       cmp    al, 8
5709 00002E25 7507                <1>       jne    short wfxl_8
5710 00002E27 B8[24580100]        <1>       mov    eax, vgafont8
5711 00002E2C EBB1                <1>       jmp    short wgfxl_0
5712                              <1> wfxl_8:
5713 00002E2E 3C0E                <1>       cmp    al, 14
5714 00002E30 7507                <1>       jne    short wfxl_9
5715 00002E32 B8[24600100]        <1>       mov    eax, vgafont14
5716 00002E37 EBA6                <1>       jmp    short wgfxl_0
5717                              <1> wfxl_9:
5718 00002E39 B8[246E0100]        <1>       mov    eax, vgafont16
5719 00002E3E EB9F                <1>       jmp    short wgfxl_0
5720                              <1>
5721                              <1> write_gfx_char_pl4:
5722                              <1>       ; 08/08/2016
5723                              <1>       ; 08/07/2016 (TRDOS 386 = TRDOS v2.0)
5724                              <1>       ;
5725                              <1>       ; derived from 'Plex86/Bochs VGABios' source code
5726                              <1>       ; vgabios-0.7a (2011)
5727                              <1>       ; by the LGPL VGABios developers Team (2001-2008)
5728                              <1>       ; 'vgabios.c', 'write_gfx_char_pl4'
5729                              <1>
5730                              <1>       ; write_gfx_char_pl4(car,attr,xcurs,ycurs,nbcols,cheight)
5731                              <1>       ; INPUT ->
```

```
                                <1>       ; AL = car, BL = attr, DL = xcurs, DH = ycurs,
                                <1>       ; [CRT_COLS] = nbcols, [CHAR_HEIGHT] = cheight
                                <1>       ; OUTPUT ->
                                <1>       ; Regen buffer updated
                                <1>
00002E40 51                     <1>       push   ecx
00002E41 53                     <1>       push   ebx
00002E42 52                     <1>       push   edx
00002E43 50                     <1>       push   eax
                                <1> wgfxpl_f0:
                                <1>       ; switch(cheight)
00002E44 8A25[BE6F0000]         <1>       mov    ah, [CHAR_HEIGHT]
00002E4A 80FC10                 <1>       cmp    ah, 16 ; case 16:
00002E4D 7507                   <1>       jne    short wgfxpl_f1
                                <1>       ; fdata = &vgafont16;
00002E4F BE[246E0100]           <1>       mov    esi, vgafont16
00002E54 EB13                   <1>       jmp    short wgfxpl_f3
                                <1> wgfxpl_f1:
00002E56 80FC0E                 <1>       cmp    ah, 14 ; case 14:
00002E59 7507                   <1>       jne    short wgfxpl_f2
00002E5B BE[24600100]           <1>       mov    esi, vgafont14
00002E60 EB07                   <1>       jmp    short wgfxpl_f3
                                <1> wgfxpl_f2:
                                <1>       ; default:
                                <1>       ;  fdata = &vgafont8;
00002E62 BE[24580100]           <1>       mov    esi, vgafont8
00002E67 B408                   <1>       mov    ah, 8
                                <1> wgfxpl_f3:
                                <1>       ; al = car
00002E69 F6E4                   <1>       mul    ah ; ah = cheight
00002E6B 25FFFF0000             <1>       and    eax, 0FFFFh ; car * cheight
                                <1>       ; src = car * cheight;
00002E70 01C6                   <1>       add    esi, eax ; esi = fdata[src+i]
                                <1>       ; addr=xcurs*8+ycurs*nbcols*64;
00002E72 88F0                   <1>       mov    al, dh ; ycurs
00002E74 8A25[BC6F0000]         <1>       mov    ah, [CRT_COLS] ; nbcols
00002E7A F6E4                   <1>       mul    ah
                                <1>       ; 08/08/2016
                                <1>       ;shl   ax, 6 ; * 64
00002E7C 66C1E003               <1>       shl    ax, 3 ; * 8
                                <1>       ;sub   dh, dh ; 0
                                <1>       ;shl   dx, 3 ; xcurs * 8
                                <1>       ;movzx edi, dx
00002E80 0FB6FA                 <1>       movzx  edi, dl
00002E83 66C1E703               <1>       shl    di, 3 ; xcurs * 8
00002E87 30F6                   <1>       xor    dh, dh
00002E89 8A15[BE6F0000]         <1>       mov    dl, [CHAR_HEIGHT]
00002E8F 66F7E2                 <1>       mul    dx
                                <1>       ; eax = ycurs*nbcols*8*[CHAR_HEIGHT]
00002E92 01C7                   <1>       add    edi, eax ; addr
00002E94 81C700000A00           <1>       add    edi, 0A0000h
                                <1>       ;
                                <1>       ; outw(VGAREG_SEQU_ADDRESS, 0x0f02);
                                <1>       ; outw(VGAREG_GRDC_ADDRESS, 0x0205);
00002E9A 66BAC403               <1>       mov    dx, 3C4h ; VGAREG_SEQU_ADDRESS
00002E9E 66B8020F               <1>       mov    ax, 0F02h
00002EA2 66EF                   <1>       out    dx, ax
00002EA4 66BACE03               <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
00002EA8 66B80502               <1>       mov    ax, 0205h
00002EAC 66EF                   <1>       out    dx, ax
                                <1>       ;
00002EAE 66BACE03               <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
00002EB2 F6C380                 <1>       test   bl, 80h ; if(attr&0x80)
00002EB5 7406                   <1>       jz     short wgfxpl_f4 ; else
                                <1>       ; outw(VGAREG_GRDC_ADDRESS, 0x1803);
00002EB7 66B80318               <1>       mov    ax, 1803h
00002EBB EB04                   <1>       jmp    short wgfxpl_f5
                                <1> wgfxpl_f4:
                                <1>       ; outw(VGAREG_GRDC_ADDRESS, 0x0003);
00002EBD 66B80300               <1>       mov    ax, 0003h
                                <1> wgfxpl_f5:
00002EC1 66EF                   <1>       out    dx, ax
                                <1>       ;
00002EC3 28FF                   <1>       sub    bh, bh ; i = 0
                                <1> wgfxpl_0:
                                <1>       ; for(i=0;i<cheight;i++)
00002EC5 57                     <1>       push   edi ; addr
00002EC6 0FB605[BC6F0000]       <1>       movzx  eax, byte [CRT_COLS] ; nbcols
00002ECD F6E7                   <1>       mul    bh ; nbcols*i
                                <1>       ; dest=addr+i*nbcols
00002ECF 01C7                   <1>       add    edi, eax ; dest
00002ED1 B580                   <1>       mov    ch, 80h ; mask = 0x80;
                                <1>       ; for(j=0;j<8;j++)
00002ED3 28C9                   <1>       sub    cl, cl ; j = 0
                                <1> wgfxpl_1:
00002ED5 D2ED                   <1>       shr    ch, cl ; mask=0x80>>j;
                                <1>       ;
                                <1>       ; outw(VGAREG_GRDC_ADDRESS, (mask << 8) | 0x08);
                                <1>       ; read_byte(0xa000,dest);
                                <1>       ;mov   dx, 3CEh ; VGAREG_GRDC_ADDRESS
00002ED7 88EC                   <1>       mov    ah, ch
00002ED9 B008                   <1>       mov    al, 8
00002EDB 66EF                   <1>       out    dx, ax
00002EDD 8A07                   <1>       mov    al, [edi] ; ? (io delay?)
                                <1>       ;
00002EDF 28C0                   <1>       sub    al, al ; attr = 0
                                <1>       ; if (fdata[src+i] & mask)
00002EE1 842E                   <1>       test   byte [esi], ch
00002EE3 7404                   <1>       jz     short wgfxpl_2  ; zf = 1
                                <1>       ; write_byte(0xa000,dest,attr&0x0f);
00002EE5 88D8                   <1>       mov    al, bl ; attr;
00002EE7 240F                   <1>       and    al, 0Fh      ; attr&0x0f
                                <1> wgfxpl_2:
                                <1>       ; write_byte(0xa000,dest,0x00);
00002EE9 8807                   <1>       mov    [edi], al ; dest (+ 0A0000h)
```

```
5837 00002EEB FEC1                <1>       inc    cl ; j++
5838 00002EED 80F908              <1>       cmp    cl, 8
5839 00002EF0 72E3                <1>       jb     short wgfxpl_1
5840 00002EF2 5F                  <1>       pop    edi
5841                              <1>       ; 08/08/2016
5842                              <1>       ;cmp   bh, 7
5843                              <1>       ;jnb   short wgfxpl_3
5844 00002EF3 FEC7                <1>       inc    bh ; i++
5845 00002EF5 3A3D[BE6F0000]      <1>       cmp    bh, [CHAR_HEIGHT]
5846 00002EFB 7303                <1>       jnb    short wgfxpl_3
5847 00002EFD 46                  <1>       inc    esi
5848 00002EFE EBC5                <1>       jmp    short wgfxpl_0
5849                              <1> wgfxpl_3:
5850                              <1>       ;mov   dx, 3CEh ; VGAREG_GRDC_ADDRESS
5851 00002F00 66B808FF            <1>       mov    ax, 0FF08h
5852 00002F04 66EF                <1>       out    dx, ax
5853 00002F06 66B80500            <1>       mov    ax, 0005h
5854 00002F0A 66EF                <1>       out    dx, ax
5855 00002F0C 66B80300            <1>       mov    ax, 0003h
5856 00002F10 66EF                <1>       out    dx, ax
5857                              <1>       ;
5858 00002F12 58                  <1>       pop    eax
5859 00002F13 5A                  <1>       pop    edx
5860 00002F14 5B                  <1>       pop    ebx
5861 00002F15 59                  <1>       pop    ecx
5862 00002F16 C3                  <1>       retn
5863                              <1>
5864                              <1> vga_write_pixel:
5865                              <1>       ; 09/07/2016 (TRDOS 386 = TRDOS v2.0)
5866                              <1>       ;
5867                              <1>       ; derived from 'Plex86/Bochs VGABios' source code
5868                              <1>       ; vgabios-0.7a (2011)
5869                              <1>       ; by the LGPL VGABios developers Team (2001-2008)
5870                              <1>       ; 'vgabios.c', 'biosfn_write_pixel'
5871                              <1>
5872                              <1>       ; INPUT ->
5873                              <1>       ;      DX = row (0-239)
5874                              <1>       ;      CX = column (0-799)
5875                              <1>       ;      AL = pixel value
5876                              <1>       ;      (AH = [CRT_MODE])
5877                              <1>       ; OUTPUT ->
5878                              <1>       ;      none
5879                              <1>
5880 00002F17 88C3                <1>       mov    bl, al ; pixel value
5881                              <1>       ;mov   ah, [CRT_MODE]
5882 00002F19 BE[D66F0000]        <1>       mov    esi, vga_modes
5883 00002F1E 89F7                <1>       mov    edi, esi
5884 00002F20 83C710              <1>       add    edi, vga_mode_count
5885                              <1> vga_wp_0:
5886 00002F23 AC                  <1>       lodsb
5887 00002F24 38E0                <1>       cmp    al, ah ; [CRT_MODE]
5888 00002F26 7405                <1>       je     short vga_wp_1
5889 00002F28 39FE                <1>       cmp    esi, edi
5890 00002F2A 72F7                <1>       jb     short vga_wp_0
5891 00002F2C C3                  <1>       retn   ; nothing to do
5892                              <1> vga_wp_1:
5893 00002F2D 83C64F              <1>       add    esi, vga_memmodel - (vga_modes + 1)
5894                              <1>       ; [ESI] = VGA memory model number (LINEAR8, PLANAR4, PLANAR1)
5895 00002F30 BF00000A00          <1>       mov    edi, 0A0000h
5896                              <1>       ;
5897 00002F35 803E04              <1>       cmp    byte [esi], PLANAR4
5898 00002F38 741D                <1>       je     short vga_wp_planar
5899 00002F3A 803E03              <1>       cmp    byte [esi], PLANAR1
5900 00002F3D 7418                <1>       je     short vga_wp_planar
5901                              <1> vga_wp_linear8:
5902                              <1>       ; addr=CX+DX*(read_word(BIOSMEM_SEG,BIOSMEM_NB_COLS)*8);
5903 00002F3F 0FB605[BC6F0000]    <1>       movzx  eax, byte [CRT_COLS] ; = [VGA_COLS] ; nbcols
5904 00002F46 66C1E003            <1>       shl    ax, 3 ; * 8
5905 00002F4A 66F7E2              <1>       mul    dx
5906 00002F4D 50                  <1>       push   eax
5907                              <1>       ;mov   edi, 0A0000h
5908 00002F4E 6601CF              <1>       add    di, cx
5909 00002F51 58                  <1>       pop    eax
5910 00002F52 01C7                <1>       add    edi, eax ; addr
5911                              <1>       ; write_byte(0xa000,addr,AL);
5912 00002F54 881F                <1>       mov    [edi], bl
5913 00002F56 C3                  <1>       retn
5914                              <1> vga_wp_planar:
5915                              <1>       ; addr = CX/8+DX*read_word(BIOSMEM_SEG,BIOSMEM_NB_COLS);
5916 00002F57 0FB7C1              <1>       movzx  eax, cx
5917 00002F5A 66C1E803            <1>       shr    ax, 3 ; CX/8
5918 00002F5E 50                  <1>       push   eax
5919 00002F5F 28E4                <1>       sub    ah, ah ; 0
5920 00002F61 A0[BC6F0000]        <1>       mov    al, [CRT_COLS] ; = [VGA_COLS] ; nbcols
5921 00002F66 66F7E2              <1>       mul    dx
5922                              <1>       ;mov   edi, 0A0000h
5923 00002F69 6601C7              <1>         add     di, ax
5924 00002F6C 58                  <1>       pop    eax
5925 00002F6D 01C7                <1>       add    edi, eax ; addr
5926 00002F6F 80E107              <1>       and    cl, 7
5927 00002F72 B580                <1>       mov    ch, 80h ; mask
5928 00002F74 D2ED                <1>       shr    ch, cl       ; mask = 0x80 >> (CX & 0x07);
5929                              <1>
5930                              <1>       ; outw(VGAREG_GRDC_ADDRESS, (mask << 8) | 0x08);
5931 00002F76 66BACE03            <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
5932 00002F7A 88EC                <1>       mov    ah, ch
5933 00002F7C B008                <1>       mov    al, 8
5934 00002F7E 66EF                <1>       out    dx, ax
5935                              <1>       ; outw(VGAREG_GRDC_ADDRESS, 0x0205);
5936 00002F80 66B80502            <1>       mov    ax, 0205h
5937 00002F84 66EF                <1>       out    dx, ax
5938                              <1>       ; data = read_byte(0xa000,addr);
5939 00002F86 8A07                <1>       mov    al, [edi] ; (delay?)
5940                              <1>       ; if (AL & 0x80)
5941                              <1>       ; {
```

```
5942                                    <1>       ;  outw(VGAREG_GRDC_ADDRESS, 0x1803);
5943                                    <1>       ; }
5944 00002F88 F6C380                   <1>       test   bl, 80h
5945 00002F8B 7406                     <1>       jz     short vga_wp_2
5946 00002F8D 66B80318                 <1>       mov    ax, 1803h
5947 00002F91 66EF                     <1>       out    dx, ax
5948                                    <1> vga_wp_2:
5949                                    <1>       ; write_byte(0xa000,addr,AL);
5950 00002F93 881F                     <1>       mov    [edi], bl
5951                                    <1>       ;
5952                                    <1>       ;mov   dx, 3CEh ; VGAREG_GRDC_ADDRESS
5953 00002F95 66B808FF                 <1>       mov    ax, 0FF08h
5954 00002F99 66EF                     <1>       out    dx, ax
5955 00002F9B 66B80500                 <1>       mov    ax, 0005h
5956 00002F9F 66EF                     <1>       out    dx, ax
5957 00002FA1 66B80300                 <1>       mov    ax, 0003h
5958 00002FA5 66EF                     <1>       out    dx, ax
5959                                    <1>       ;
5960 00002FA7 C3                       <1>       retn
5961                                    <1>
5962                                    <1> vga_read_pixel:
5963                                    <1>       ; 09/07/2016 (TRDOS 386 = TRDOS v2.0)
5964                                    <1>       ;
5965                                    <1>       ; derived from 'Plex86/Bochs VGABios' source code
5966                                    <1>       ; vgabios-0.7a (2011)
5967                                    <1>       ; by the LGPL VGABios developers Team (2001-2008)
5968                                    <1>       ; 'vgabios.c', 'biosfn_read_pixel'
5969                                    <1>
5970                                    <1>       ; INPUT ->
5971                                    <1>       ;      DX = row (0-239)
5972                                    <1>       ;      CX = column (0-799)
5973                                    <1>       ;      (AH = [CRT_MODE])
5974                                    <1>       ; OUTPUT ->
5975                                    <1>       ;      AL = pixel value
5976                                    <1>
5977                                    <1>       ;mov   ah, [CRT_MODE]
5978 00002FA8 BE[D66F0000]             <1>       mov    esi, vga_modes
5979 00002FAD 89F7                     <1>       mov    edi, esi
5980 00002FAF 83C710                   <1>       add    edi, vga_mode_count
5981                                    <1> vga_rp_0:
5982 00002FB2 AC                       <1>       lodsb
5983 00002FB3 38E0                     <1>       cmp    al, ah ; [CRT_MODE]
5984 00002FB5 7405                     <1>       je     short vga_rp_1
5985 00002FB7 39FE                     <1>       cmp    esi, edi
5986 00002FB9 72F7                     <1>       jb     short vga_rp_0
5987 00002FBB C3                       <1>       retn   ; nothing to do
5988                                    <1> vga_rp_1:
5989 00002FBC 83C64F                   <1>       add    esi, vga_memmodel - (vga_modes + 1)
5990                                    <1>       ; [ESI] = VGA memory model number (LINEAR8, PLANAR4, PLANAR1)
5991 00002FBF BF00000A00               <1>       mov    edi, 0A0000h
5992                                    <1>       ;
5993 00002FC4 803E04                   <1>       cmp    byte [esi], PLANAR4
5994 00002FC7 741D                     <1>       je     short vga_rp_planar
5995 00002FC9 803E03                   <1>       cmp    byte [esi], PLANAR1
5996 00002FCC 7418                     <1>       je     short vga_rp_planar
5997                                    <1> vga_rp_linear8:
5998                                    <1>       ; addr=CX+DX*(read_word(BIOSMEM_SEG,BIOSMEM_NB_COLS)*8);
5999 00002FCE 0FB605[BC6F0000]         <1>       movzx  eax, byte [CRT_COLS] ; = [VGA_COLS] ; nbcols
6000 00002FD5 66C1E003                 <1>       shl    ax, 3 ; * 8
6001 00002FD9 66F7E2                   <1>       mul    dx
6002 00002FDC 50                       <1>       push   eax
6003                                    <1>       ;mov   edi, 0A0000h
6004 00002FDD 6601CF                   <1>       add    di, cx
6005 00002FE0 58                       <1>       pop    eax
6006 00002FE1 01C7                     <1>       add    edi, eax ; addr
6007                                    <1>       ; attr=read_byte(0xa000,addr);
6008 00002FE3 8A07                     <1>       mov    al, [edi] ; pixel value
6009 00002FE5 C3                       <1>       retn
6010                                    <1> vga_rp_planar:
6011                                    <1>       ; addr = CX/8+DX*read_word(BIOSMEM_SEG,BIOSMEM_NB_COLS);
6012 00002FE6 0FB7C1                   <1>       movzx  eax, cx
6013 00002FE9 66C1E803                 <1>       shr    ax, 3 ; CX/8
6014 00002FED 50                       <1>       push   eax
6015 00002FEE 28E4                     <1>       sub    ah, ah ; 0
6016 00002FF0 A0[BC6F0000]             <1>       mov    al, [CRT_COLS] ; = [VGA_COLS] ; nbcols
6017 00002FF5 66F7E2                   <1>       mul    dx
6018                                    <1>       ;mov   edi, 0A0000h
6019 00002FF8 6601C7                   <1>         add     di, ax
6020 00002FFB 58                       <1>       pop    eax
6021 00002FFC 01C7                     <1>       add    edi, eax ; addr
6022 00002FFE 80E107                   <1>       and    cl, 7
6023 00003001 B580                     <1>       mov    ch, 80h ; mask
6024 00003003 D2ED                     <1>       shr    ch, cl       ; mask = 0x80 >> (CX & 0x07);
6025                                    <1>       ; attr = 0x00;
6026 00003005 30DB                     <1>       xor    bl, bl ; attr = bl = 0,
6027 00003007 30C9                     <1>       xor    cl, cl ; i = cl = 0
6028                                    <1>       ; for(i=0;i<4;i++)
6029                                    <1>             ; {
6030                                    <1>             ;  outw(VGAREG_GRDC_ADDRESS, (i << 8) | 0x04);
6031                                    <1>             ;  data = read_byte(0xa000,addr) & mask;
6032                                    <1>             ;  if (data > 0) attr |= (0x01 << i);
6033                                    <1>             ; }
6034                                    <1> vga_rp_2:
6035 00003009 88CC                     <1>       mov    ah, cl ; i << 8
6036 0000300B B004                     <1>       mov    al, 4  ; | 0x04
6037 0000300D 66BACE03                 <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
6038 00003011 66EF                     <1>       out    dx, ax
6039                                    <1>       ; data = read_byte(0xa000,addr) & mask;
6040 00003013 8A07                     <1>       mov    al, [edi]
6041 00003015 20E8                     <1>       and    al, ch ; & mask
6042                                    <1>       ; if (data > 0) attr |= (0x01 << i);
6043 00003017 08C0                     <1>       or     al, al
6044 00003019 7408                     <1>       jz     short vga_rp_3 ; al = 0
6045 0000301B B701                     <1>       mov    bh, 1
6046 0000301D D2E7                     <1>       shl    bh, cl ; (0x01 << i)
```

```
6047 0000301F 08FB              <1>       or    bl, bh ; attr |= (0x01 << i)
6048 00003021 88D8              <1>       mov   al, bl ; pixel value
6049                            <1> vga_rp_3:
6050 00003023 C3                <1>       retn
6051                            <1>
6052                            <1> vga_beeper:
6053                            <1>       ; 04/08/2016  (TRDOS 386 = TRDOS v2.0)
6054 00003024 FB                <1>       sti
6055                            <1>       ;mov   bh, [ACTIVE_PAGE]
6056 00003025 E9C0F3FFFF        <1>         jmp     beeper_gfx
6057                            <1>
6058                            <1> vga_write_teletype:
6059                            <1>       ; 09/12/2017
6060                            <1>       ; 06/08/2016
6061                            <1>       ; 04/08/2016
6062                            <1>       ; 01/08/2016
6063                            <1>       ; 31/07/2016
6064                            <1>       ; 09/07/2016 (TRDOS 386 = TRDOS v2.0)
6065                            <1>       ;
6066                            <1>       ; derived from 'Plex86/Bochs VGABios' source code
6067                            <1>       ; vgabios-0.7a (2011)
6068                            <1>       ; by the LGPL VGABios developers Team (2001-2008)
6069                            <1>       ; 'vgabios.c', 'biosfn_write_teletype'
6070                            <1>       ; 'biosfn_write_char_only'
6071                            <1>
6072                            <1>       ; INPUT ->
6073                            <1>       ; [CRT_MODE] = current video mode (>7)
6074                            <1>       ; AL = Character to write
6075                            <1>       ; BL = Color of character
6076                            <1>       ; OUTPUT ->
6077                            <1>       ; Regen buffer updated
6078                            <1>
6079                            <1>       ; biosfn_write_teletype (car, page, attr, flag)
6080                            <1>       ; car = character (AL)
6081                            <1>       ; page = 0
6082                            <1>       ; attr = color (BL)
6083                            <1>       ; 'flag' not used
6084                            <1>
6085 0000302A 8A25[BA6F0000]    <1>       mov   ah, [CRT_MODE]
6086 00003030 88C7              <1>       mov   bh, al ; character
6087 00003032 668B15[AE810100]  <1>       mov   dx, [CURSOR_POSN] ; cursor pos for page 0
6088                            <1>
6089 00003039 BE[DE6F0000]      <1>       mov   esi, vga_g_modes
6090 0000303E 89F7              <1>       mov   edi, esi
6091 00003040 83C708            <1>       add   edi, vga_g_mode_count
6092                            <1> vga_wtty_0:
6093 00003043 AC                <1>       lodsb
6094 00003044 38E0              <1>       cmp   al, ah ; [CRT_MODE]
6095 00003046 7405              <1>       je    short vga_wtty_2
6096 00003048 39FE              <1>       cmp   esi, edi
6097 0000304A 72F7              <1>       jb    short vga_wtty_0
6098                            <1> vga_wtty_1:
6099 0000304C C3                <1>       retn  ; nothing to do
6100                            <1> vga_wtty_2:
6101 0000304D 80FF07            <1>       cmp   bh, 07h ; bell (beep)
6102 00003050 74D2              <1>       je    short vga_beeper  ; u11
6103 00003052 80FF08            <1>       cmp   bh, 08h ; backspace
6104 00003055 7508              <1>       jne   short vga_wtty_3
6105                            <1>       ; if(xcurs>0)xcurs--;
6106 00003057 08D2              <1>       or    dl, dl ; xcurs (column)
6107 00003059 74F1              <1>       jz    short vga_wtty_1
6108 0000305B FECA              <1>       dec   dl ; xcurs--;
6109 0000305D EB59              <1>         jmp     short vga_wtty_12
6110                            <1> vga_wtty_3:
6111 0000305F 80FF0D            <1>       cmp   bh, 0Dh ; carriage return (\r)
6112 00003062 7504              <1>       jne   short vga_wtty_4
6113                            <1>       ; xcurs=0;
6114 00003064 28D2              <1>       sub   dl, dl ; 0
6115 00003066 EB50              <1>         jmp     short vga_wtty_12
6116                            <1> vga_wtty_4:
6117 00003068 80FF0A            <1>       cmp   bh, 0Ah ; new line (\n)
6118 0000306B 7504              <1>       jne   short vga_wtty_5
6119                            <1>       ; ycurs++;
6120 0000306D FEC6              <1>       inc   dh ; next row
6121 0000306F EB62              <1>         jmp     short vga_wtty_11
6122                            <1> vga_wtty_5:
6123 00003071 80FF09            <1>       cmp   bh, 09h ; tab stop
6124 00003074 7527              <1>       jne   short vga_wtty_8
6125 00003076 88D0              <1>       mov   al, dl
6126                            <1>       ;cbw
6127 00003078 30E4              <1>       xor   ah, ah ; 09/12/2017
6128 0000307A B108              <1>       mov   cl, 8
6129 0000307C F6F1              <1>       div   cl
6130 0000307E 28E1              <1>       sub   cl, ah
6131                            <1>       ;
6132 00003080 B720              <1>       mov   bh, 20h ; space
6133                            <1> vga_wtty_6: ; tab stop loop
6134 00003082 6651              <1>       push  cx
6135 00003084 6653              <1>       push  bx
6136 00003086 E812000000        <1>       call  vga_wtty_8
6137 0000308B 665B              <1>       pop   bx  ; bh = character, bl = color
6138 0000308D 6659              <1>       pop   cx
6139 0000308F FEC9              <1>       dec   cl
6140 00003091 7409              <1>       jz    short vga_wtty_7
6141 00003093 668B15[AE810100]  <1>       mov   dx, [CURSOR_POSN] ; new cursor position (pg 0)
6142 0000309A EBE6              <1>       jmp   short vga_wtty_6
6143                            <1> vga_wtty_7:
6144 0000309C C3                <1>       retn
6145                            <1>       ;
6146                            <1> vga_wtty_8:
6147 0000309D 83C64F            <1>       add   esi, vga_g_memmodel - (vga_g_modes + 1)
6148                            <1>       ; [ESI] = VGA memory model number (LINEAR8, PLANAR4, PLANAR1)
6149 000030A0 BF00000A00        <1>       mov   edi, 0A0000h
6150                            <1>       ;
6151 000030A5 88F8              <1>       mov   al, bh ; character
```

```
6152                                    <1>       ;
6153 000030A7 803E04                    <1>       cmp    byte [esi], PLANAR4
6154 000030AA 7414                      <1>       je     short vga_wtty_planar
6155 000030AC 803E03                    <1>       cmp    byte [esi], PLANAR1
6156 000030AF 740F                      <1>       je     short vga_wtty_planar
6157                                    <1> vga_wtty_linear8:
6158                                    <1>       ; write_gfx_char_lin(car,attr,xcurs,ycurs,nbcols);
6159                                    <1>       ; AL = car, BL = attr (color), DL = xcurs, DH = ycurs,
6160                                    <1>       ; [CRT_COLS] = nbcols
6161 000030B1 E8E7FCFFFF                <1>       call   write_gfx_char_lin
6162 000030B6 EB0D                      <1>       jmp    short vga_wtty_9
6163                                    <1>
6164                                    <1> vga_wtty_12:
6165                                    <1>       ; 09/07/2016
6166                                    <1>       ; set cursor position
6167                                    <1>       ; NOTE: Hardware cursor position will not be set
6168                                    <1>       ;   in any VGA modes (>7)
6169                                    <1>       ;   But, cursor position will be saved into
6170                                    <1>       ;   [CURSOR_POSN].
6171                                    <1>       ;   TRDOS 386 (TRDOS v2.0) uses only one page
6172                                    <1>       ;   (page 0) for all graphics modes.
6173                                    <1>
6174 000030B8 668915[AE810100]          <1>       mov    [CURSOR_POSN], dx ; save cursor pos for pg 0
6175                                    <1>       ; 04/08/2016
6176                                    <1>       ;mov   bh, [ACTIVE_PAGE] ; = 0
6177                                    <1>       ;call  _set_cpos
6178 000030BF C3                       <1>       retn
6179                                    <1>
6180                                    <1> vga_wtty_planar:
6181                                    <1>       ; write_gfx_char_pl4(car,attr,xcurs,ycurs,nbcols,cheight);
6182                                    <1>       ; AL = car, BL = attr (color), DL = xcurs, DH = ycurs,
6183                                    <1>       ; [CRT_COLS]= nbcols, [CHAR_HEIGHT] = cheight
6184 000030C0 E87BFDFFFF                <1>       call   write_gfx_char_pl4
6185                                    <1> vga_wtty_9:
6186 000030C5 FEC2                      <1>       inc    dl ; xcurs++;
6187                                    <1> vga_wtty_10:
6188                                    <1>       ; Do we need to wrap ?
6189                                    <1>       ; if(xcurs==nbcols)
6190 000030C7 3A15[BC6F0000]            <1>       cmp    dl, [CRT_COLS] ; [VGA_COLS]
6191 000030CD 7204                      <1>       jb     short vga_wtty_11 ; no
6192 000030CF 28D2                      <1>       sub    dl, dl ; xcurs=0;
6193 000030D1 FEC6                      <1>       inc    dh ;  ycurs++;
6194                                    <1> vga_wtty_11:
6195                                    <1>       ; Do we need to scroll ?
6196                                    <1>       ; if(ycurs==nbrows)
6197 000030D3 3A35[C26F0000]            <1>       cmp    dh, [VGA_ROWS]
6198 000030D9 72DD                      <1>       jb     short vga_wtty_12 ; no
6199                                    <1>       ;
6200                                    <1>       ; biosfn_scroll (nblines,attr,rul,cul,rlr,clr,page,dir)
6201                                    <1>       ; al = nblines = 1, bl = attr (color) = 0
6202                                    <1>       ; ch = rul, cl = cul, dh = rlr, dl = clr, page = 0
6203                                    <1>       ; dir = SCROLL_UP
6204                                    <1>
6205 000030DB B001                      <1>       mov    al, 1
6206 000030DD 28DB                      <1>       sub    bl, bl ; 0 ; blank/black line (attr=0) will be used
6207 000030DF 6629C9                    <1>       sub    cx, cx ; 0,0
6208                                    <1>
6209                                    <1>       ; 06/08/2016
6210 000030E2 8A35[C26F0000]            <1>       mov    dh, [VGA_ROWS]
6211 000030E8 FECE                      <1>       dec    dh ; nbrows -1
6212                                    <1>
6213 000030EA 6652                      <1>       push   dx     ; 04/08/2016
6214 000030EC 8A15[BC6F0000]            <1>       mov    dl, [CRT_COLS]
6215 000030F2 FECA                      <1>       dec    dl ; nbcols -1
6216                                    <1>
6217 000030F4 8A25[BA6F0000]            <1>       mov    ah, [CRT_MODE]
6218                                    <1>
6219                                    <1>       ; biosfn_scroll(0x01,0x00,0,0,nbrows-1,nbcols-1,page,SCROLL_UP);
6220 000030FA E8FBF4FFFF                <1>       call   vga_graphics_up
6221                                    <1>       ; 04/08/2016
6222 000030FF 665A                      <1>       pop    dx
6223                                    <1>       ;dec   dh ; ycurs-=1
6224 00003101 EBB5                      <1>       jmp    short vga_wtty_12
6225                                    <1>
6226                                    <1> font_setup:
6227                                    <1>       ; 09/01/2021 (TRDOS 386 v2.0.3)
6228                                    <1>       ; 09/07/2016
6229                                    <1>       ; character generator (font loading) functions
6230                                    <1>       ;
6231                                    <1>       ; derived from 'Plex86/Bochs VGABios' source code
6232                                    <1>       ; vgabios-0.7a (2011)
6233                                    <1>       ; by the LGPL VGABios developers Team (2001-2008)
6234                                    <1>       ; 'vgabios.c', 'int10_func'
6235                                    <1>
6236                                    <1>       ; AX = 1100H ; Load User-Defined Font (EGA/VGA)
6237                                    <1>       ;
6238                                    <1>         ; BH    height of each character (bytes per character definition)
6239                                    <1>         ; (BL   font block to load (EGA: 0-3; VGA: 0-7))
6240                                    <1>       ; CX    number of characters to redefine (<=256)
6241                                    <1>         ; DX    ASCII code of the first character defined at ES:BP
6242                                    <1>         ; EBP      address of font-definition information
6243                                    <1>       ;      (in user's memory space)
6244                                    <1>
6245                                    <1>       ; case 0x11:
6246                                    <1>       ; switch(GET_AL())
6247                                    <1>            ; {
6248                                    <1>       ; case 0x00:
6249                                    <1>         ; case 0x10:
6250                                    <1>         ; biosfn_load_text_user_pat(GET_AL(),ES,BP,CX,DX,GET_BL(),GET_BH());
6251                                    <1>         ; break;
6252                                    <1>
6253                                    <1>       ; AX = 1110H ; Load and Activate User-Defined Font (EGA/VGA)
6254 00003103 08C0                      <1>       or     al, al ; 0
6255 00003105 7404                      <1>       jz     short font_setup_0
6256 00003107 3C10                      <1>       cmp    al, 10h
```

```
6257 00003109 7511                <1>       jne    short font_setup_1
6258                              <1> font_setup_0:
6259 0000310B E8CE000000          <1>       call   transfer_user_fonts
6260 00003110 721C                <1>       jc     short font_setup_error
6261 00003112 E8B2010000          <1>       call   load_text_user_pat
6262 00003117 E93BEAFFFF          <1>         jmp     VIDEO_RETURN
6263                              <1> font_setup_1:
6264                              <1>       ; AX = 1101H ; Load ROM 8x14 Character Set (EGA/VGA)
6265                              <1>       ; case 0x01:
6266                              <1>         ; case 0x11:
6267                              <1>         ; biosfn_load_text_8_14_pat(GET_AL(),GET_BL());
6268                              <1>         ; break;
6269 0000311C 3C01                <1>       cmp    al, 1
6270 0000311E 7404                <1>       je     short font_setup_2
6271 00003120 3C11                <1>       cmp    al, 11h
6272 00003122 7511                <1>       jne    short font_setup_3
6273                              <1> font_setup_2:
6274                              <1>       ; AX = 1111H ; Load and Activate ROM 8x14 Character Set (EGA/VGA)
6275                              <1>       ; (BL = font block to load (EGA: 0-3; VGA: 0-7))
6276 00003124 E8DC020000          <1>       call   load_text_8_14_pat
6277 00003129 E929EAFFFF          <1>         jmp     VIDEO_RETURN
6278                              <1> font_setup_error:
6279 0000312E 29C0                <1>       sub    eax, eax ; 0 -> fonts could not be loaded
6280 00003130 E927EAFFFF          <1>       jmp    _video_return
6281                              <1> font_setup_3:
6282                              <1>       ; AX = 1102H ; Load ROM 8x8 Character Set (EGA/VGA)
6283                              <1>       ; case 0x02:
6284                              <1>         ; case 0x12:
6285                              <1>         ; biosfn_load_text_8_8_pat(GET_AL(),GET_BL());
6286                              <1>         ; break;
6287 00003135 3C02                <1>       cmp    al, 2
6288 00003137 7404                <1>       je     short font_setup_4
6289 00003139 3C12                <1>       cmp    al, 12h
6290 0000313B 750A                <1>       jne    short font_setup_5
6291                              <1> font_setup_4:
6292                              <1>       ; AX = 1112H ; Load and Activate ROM 8x8 Character Set (EGA/VGA)
6293                              <1>       ; (BL = font block to load (EGA: 0-3; VGA: 0-7))
6294 0000313D E8F3020000          <1>       call   load_text_8_8_pat
6295 00003142 E910EAFFFF          <1>         jmp     VIDEO_RETURN
6296                              <1> font_setup_5:
6297                              <1>       ; AX = 1104H ; Load ROM 8x16 Character Set (EGA/VGA)
6298                              <1>       ; case 0x04:
6299                              <1>         ; case 0x14:
6300                              <1>         ; biosfn_load_text_8_16_pat(GET_AL(),GET_BL());
6301                              <1>         ; break;
6302 00003147 3C04                <1>       cmp    al, 4
6303 00003149 7404                <1>       je     short font_setup_6
6304 0000314B 3C14                <1>       cmp    al, 14h
6305 0000314D 750A                <1>       jne    short font_setup_7
6306                              <1> font_setup_6:
6307                              <1>       ; AX = 1114H ; Load and Activate ROM 8x16 Character Set (EGA/VGA)
6308                              <1>       ; (BL = font block to load (EGA: 0-3; VGA: 0-7))
6309 0000314F E82A030000          <1>       call   load_text_8_16_pat
6310 00003154 E9FEE9FFFF          <1>         jmp     VIDEO_RETURN
6311                              <1> font_setup_7:
6312                              <1>       ; Note: AX=1120h (Setup INT 1Fh, EXT_PTR) is not needed
6313                              <1>       ; for TRDOS 386 (TRDOS v2.0) video functionality;
6314                              <1>       ; because, originally EXT_PTR (font address) was used for
6315                              <1>       ; chars 80h to 0FFh (after the first 128 ASCII char fonts), for
6316                              <1>       ; CGA graphics mode; currenty, 'vgafont8' address has 256 chars!
6317                              <1>       ;
6318                              <1>       ; case 0x20:
6319                              <1>         ; biosfn_load_gfx_8_8_chars(ES,BP);
6320                              <1>         ; break;
6321                              <1>       ; case 0x21:
6322                              <1>         ; biosfn_load_gfx_user_chars(ES,BP,CX,GET_BL(),GET_DL());
6323                              <1>         ; break;
6324                              <1>       ; AX = 1121H ; Setup User-Defined Font for Graphics Mode (VGA)
6325                              <1>       ; BL   screen rows code: 00H = user-specified (in DL)
6326                              <1>         ;                        01H = 14 rows
6327                              <1>         ;                        02H = 25 rows
6328                              <1>         ;                        03H = 43 rows
6329                              <1>         ; CX   bytes per character definition
6330                              <1>         ; DL   (when BL=0) custom number of character rows on screen
6331                              <1>         ; EBP  address of font-definition information (user's mem space)
6332                              <1>
6333 00003159 3C21                <1>       cmp    al, 21h
6334 0000315B 7531                <1>       jne    short font_setup_9
6335                              <1>
6336                              <1>       ; TRDOS 386 modification !
6337                              <1>       ; dh = 0 -> 256 characters
6338                              <1>       ; dh = 80h -> second 128 characters
6339                              <1>       ; dh = 0FFh -> first 128 characters
6340                              <1>
6341                              <1>       ; 09/01/2021 (TRDOS 386 v2.0.3)
6342                              <1>       ;push  ebx
6343 0000315D 51                  <1>       push   ecx
6344 0000315E 52                  <1>       push   edx
6345 0000315F 30D2                <1>       xor    dl, dl
6346 00003161 88CF                <1>       mov    bh, cl ; character height
6347 00003163 66B90001            <1>       mov    cx, 100h ; 256
6348 00003167 08F6                <1>       or     dh, dh ; 0
6349 00003169 7410                <1>       jz     short  font_setup_8
6350 0000316B FECD                <1>       dec    ch ; cx = 0
6351 0000316D 80FEFF              <1>       cmp    dh, 0FFh
6352 00003170 7409                <1>       je     short font_setup_8 ; 1st 128 chars
6353                              <1>       ; 2nd 128 chars
6354 00003172 80FE80              <1>       cmp    dh, 80h
6355 00003175 75B7                <1>       jne    short font_setup_error ; invalid !
6356 00003177 88F1                <1>       mov    cl, dh
6357 00003179 86D6                <1>       xchg   dl, dh
6358                              <1>       ; number of chars, cx = 80h
6359                              <1>       ; start char, dl = 80h
6360                              <1> font_setup_8:
6361 0000317B E85E000000          <1>       call   transfer_user_fonts
```

```
6362 00003180 5A                <1>       pop    edx
6363 00003181 59                <1>       pop    ecx
6364                            <1>       ;pop   ebx
6365 00003182 72AA              <1>       jc     short font_setup_error
6366                            <1>       ; ebp = user's font data address in system's memory space
6367 00003184 E83F030000        <1>       call   load_gfx_user_chars
6368 00003189 E9C9E9FFFF        <1>         jmp     VIDEO_RETURN
6369                            <1> font_setup_9:
6370                            <1>       ; case 0x22:
6371                            <1>         ; biosfn_load_gfx_8_14_chars(GET_BL());
6372                            <1>         ; break;
6373 0000318E 3C22              <1>       cmp    al, 22h
6374 00003190 750A              <1>       jne    short font_setup_10
6375 00003192 E86D030000        <1>       call   load_gfx_8_14_chars
6376 00003197 E9BBE9FFFF        <1>         jmp     VIDEO_RETURN
6377                            <1> font_setup_10:
6378                            <1>       ; case 0x23:
6379                            <1>         ; biosfn_load_gfx_8_8_dd_chars(GET_BL());
6380                            <1>         ; break;
6381 0000319C 3C23              <1>       cmp    al, 23h
6382 0000319E 750A              <1>       jne    short font_setup_11
6383 000031A0 E8A0030000        <1>       call   load_gfx_8_8_chars
6384 000031A5 E9ADE9FFFF        <1>         jmp     VIDEO_RETURN
6385                            <1> font_setup_11:
6386                            <1>       ; case 0x24:
6387                            <1>         ; biosfn_load_gfx_8_16_chars(GET_BL());
6388                            <1>         ; break;
6389 000031AA 3C24              <1>       cmp    al, 24h
6390 000031AC 750A              <1>       jne    short font_setup_12
6391 000031AE E8D3030000        <1>       call   load_gfx_8_16_chars
6392 000031B3 E99FE9FFFF        <1>         jmp     VIDEO_RETURN
6393                            <1> font_setup_12:
6394                            <1>       ; case 0x30:
6395                            <1>         ; biosfn_get_font_info(GET_BH(),&ES,&BP,&CX,&DX);
6396                            <1>         ; break;
6397 000031B8 3C30              <1>       cmp    al, 30h
6398 000031BA 750A              <1>       jne    short font_setup_13
6399 000031BC E806040000        <1>       call   get_font_info
6400                            <1>       ; eax = return value (info: 4 bytes for 4 parms)
6401                            <1>       ; eax = 0 -> invalid function (input)
6402 000031C1 E996E9FFFF        <1>         jmp     _video_return
6403                            <1> font_setup_13:
6404 000031C6 3C03              <1>       cmp    al, 03h ; AX = 1103h
6405 000031C8 750D              <1>       jne    short font_setup_14
6406                            <1>       ; biosfn_set_text_block_specifier:
6407                            <1>       ; BL = font block selector code
6408                            <1>       ; NOTE: TRDOS 386 only uses and sets font block 0
6409                            <1>       ; (It is as BL = 0 for TRDOS 386)
6410 000031CA 66BAC403          <1>       mov    dx, 3C4h ; VGAREG_SEQU_ADDRESS
6411                            <1>       ;mov   ah, bl
6412 000031CE 28E4              <1>       sub    ah, ah ; 0
6413                            <1>       ;mov   al, 03h
6414 000031D0 66EF              <1>       out    dx, ax
6415 000031D2 E980E9FFFF        <1>       jmp     VIDEO_RETURN
6416                            <1>
6417                            <1> font_setup_14:
6418 000031D7 29C0              <1>       sub    eax, eax ; 0 = invalid function
6419 000031D9 E97EE9FFFF        <1>         jmp     _video_return
6420                            <1>
6421                            <1> transfer_user_fonts:
6422                            <1>       ; 19/01/2021
6423                            <1>       ; 09/01/2021
6424                            <1>       ; 05/01/2021 (TRDOS 386 v2.0.3)
6425                            <1>
6426                            <1>       ; BH    height of each character (bytes per character)
6427                            <1>       ; CX    number of characters to redefine (<=256)
6428                            <1>         ; DX    ASCII code of the first character defined at EBP
6429                            <1>         ; EBP      address of font-definition information
6430                            <1>       ;      (in user's memory space)
6431                            <1>
6432                            <1>       ; Modified registers: eax, edx, ecx, esi, edi, ebp
6433                            <1>       ;
6434                            <1>       ; output:
6435                            <1>       ;      ebp = user font data address in system memory
6436                            <1>
6437 000031DE 81E2FFFF0000     <1>       and    edx, 0FFFFh
6438 000031E4 81E1FFFF0000     <1>       and    ecx, 0FFFFh
6439 000031EA 7537              <1>       jnz    short transfer_user_fonts_5
6440                            <1>
6441 000031EC 09D2              <1>       or     edx, edx
6442 000031EE 7531              <1>       jnz    short transfer_user_fonts_4
6443 000031F0 09ED              <1>       or     ebp, ebp
6444 000031F2 752D              <1>       jnz    short transfer_user_fonts_4
6445                            <1>
6446                            <1>       ; cx = 0, dx = 0, ebp = 0
6447                            <1>       ; copy system font to user font
6448                            <1>
6449 000031F4 B140              <1>       mov    cl, 64 ; 64 dwords
6450                            <1>
6451 000031F6 80FF10            <1>       cmp    bh, 16
6452 000031F9 7417              <1>       je     short transfer_user_fonts_2
6453 000031FB 80FF08            <1>       cmp    bh, 8
6454 000031FE 7406              <1>       je     short transfer_user_fonts_1
6455                            <1>
6456 00003200 BD[24600100]      <1>       mov    ebp, vgafont14
6457 00003205 C3                <1>       retn
6458                            <1>
6459                            <1> transfer_user_fonts_1:
6460 00003206 BF00500900        <1>       mov    edi, VGAFONT8USER
6461 0000320B BE[24580100]      <1>       mov    esi, vgafont8
6462 00003210 EB0A              <1>       jmp    short transfer_user_fonts_3
6463                            <1>
6464                            <1> transfer_user_fonts_2:
6465 00003212 BF00400900        <1>       mov    edi, VGAFONT16USER
6466 00003217 BE[246E0100]      <1>       mov    esi, vgafont16
```

```
                                <1> transfer_user_fonts_3:
0000321C 89FD                   <1>       mov    ebp, edi
0000321E F3A5                   <1>       rep    movsd
00003220 C3                     <1>       retn
                                <1>
                                <1> transfer_user_fonts_4:
00003221 F9                     <1>       stc
00003222 C3                     <1>       retn
                                <1>
                                <1> transfer_user_fonts_5:
00003223 09ED                   <1>       or     ebp, ebp
00003225 74FA                   <1>       jz     short transfer_user_fonts_4 ; invalid address !
                                <1>
00003227 6681F90001             <1>       cmp    cx, 256
0000322C 77F3                   <1>       ja     short transfer_user_fonts_4
0000322E 29D1                   <1>       sub    ecx, edx
00003230 76EF                   <1>       jna    short transfer_user_fonts_4
                                <1>
00003232 80FF0E                 <1>       cmp    bh, 14 ; 8x14 font
                                <1>                     ; (there is not an alternative buffer)
00003235 7525                   <1>       jne    short transfer_user_fonts_6
                                <1>
                                <1>       ; use system's 8x14 font space if permission flag is 1
00003237 F605[82120300]80       <1>       test   byte [ufont], 80h
0000323E 74E1                   <1>       jz     short transfer_user_fonts_4 ; not allowed
                                <1>
                                <1>       ; permission is given (for vgafont14 location etc.)
                                <1>       ; (for permanent font modification)
                                <1>       ;
                                <1>       ; 19/01/2021
                                <1>       ; Note: Permission flag can be set by 'root' while
                                <1>       ;     system is not in multi tasking/user mode
                                <1>       ;     while [multi_tasking] = 0 and [u.uid] = 0
                                <1>
00003240 52                     <1>       push   edx
00003241 30E4                   <1>       xor    ah, ah
00003243 88F8                   <1>       mov    al, bh ; mov al, 14
00003245 66F7E1                 <1>       mul    cx
                                <1>              ; ecx = byte count
00003248 89C1                   <1>       mov    ecx, eax
0000324A 5A                     <1>       pop    edx
0000324B 30E4                   <1>       xor    ah, ah
0000324D 88F8                   <1>       mov    al, bh ; mov ax, 14 ; bytes per character
0000324F 66F7E2                 <1>       mul    dx
00003252 6689C2                 <1>       mov    dx, ax ; char offset
00003255 BF[24600100]           <1>       mov    edi, vgafont14
0000325A EB4C                   <1>       jmp    short transfer_user_fonts_8
                                <1> transfer_user_fonts_6:
0000325C 80FF08                 <1>       cmp    bh, 8 ; 8x8 font
0000325F 7522                   <1>       jne    short transfer_user_fonts_7 ; 8x16 font
00003261 66C1E203               <1>       shl    dx, 3 ; * 8
00003265 66C1E103               <1>       shl    cx, 3 ; * 8
                                <1>       ; 09/01/2021
00003269 BF00500900             <1>       mov    edi, VGAFONT8USER
0000326E F605[82120300]08       <1>       test   byte [ufont], 8  ; already loaded ?
00003275 7531                   <1>       jnz    short transfer_user_fonts_8 ; yes
00003277 BE[24580100]           <1>       mov    esi, vgafont8
0000327C E83B000000             <1>       call   transfer_user_fonts_10
00003281 EB25                   <1>       jmp    short transfer_user_fonts_8
                                <1> transfer_user_fonts_7:
00003283 80FF10                 <1>       cmp    bh, 16 ; 8x16 font
00003286 7599                   <1>       jne    short transfer_user_fonts_4  ; invalid !
00003288 66C1E204               <1>       shl    dx, 4 ; * 16
0000328C 66C1E104               <1>       shl    cx, 4 ; * 16
00003290 BF00400900             <1>       mov    edi, VGAFONT16USER
00003295 F605[82120300]10       <1>       test   byte [ufont], 16  ; already loaded ?
0000329C 750A                   <1>       jnz    short transfer_user_fonts_8 ; yes
0000329E BE[246E0100]           <1>       mov    esi, vgafont16
000032A3 E814000000             <1>       call   transfer_user_fonts_10
                                <1> transfer_user_fonts_8:
000032A8 01D7                   <1>       add    edi, edx ; char offset
                                <1>       ; 09/07/2016
                                <1>       ;and   ecx, 0FFFFh
                                <1>       ; ECX = byte count
                                <1>       ;push  ecx
000032AA 89EE                   <1>       mov    esi, ebp ; user's font buffer
                                <1>       ; 09/01/2021
000032AC 89FD                   <1>       mov    ebp, edi ; system addr for user's font
                                <1>       ; 05/01/2021
                                <1>       ;mov   edi, Cluster_Buffer  ; system buffer
000032AE E85BE00000             <1>       call   transfer_from_user_buffer
                                <1>       ;pop   ecx
                                <1>       ; ecx = transfer (byte) count = character count
000032B3 7206                   <1>       jc     short transfer_user_fonts_9
                                <1>       ; 05/01/2021
                                <1>       ;mov   ebp, Cluster_Buffer
                                <1>
000032B5 083D[82120300]         <1>       or     byte [ufont], bh
                                <1>                     ; 8x8 or 8x16 user font ready
                                <1> transfer_user_fonts_9:
000032BB C3                     <1>       retn
                                <1>
                                <1> transfer_user_fonts_10:
                                <1>       ; 09/01/2021
000032BC 56                     <1>       push   esi
000032BD 57                     <1>       push   edi
000032BE 51                     <1>       push   ecx
000032BF 66B94000               <1>       mov    cx, 64
000032C3 F3A5                   <1>       rep    movsd
000032C5 59                     <1>       pop    ecx
000032C6 5F                     <1>       pop    edi
000032C7 5E                     <1>       pop    esi
000032C8 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>
000032C9 50             <1>       push   eax
000032CA E83C000000     <1>       call   get_font_access
000032CF 28DB           <1>       sub    bl, bl ; i = 0
                        <1> ltup_1:
000032D1 88D8           <1>       mov    al, bl
000032D3 F6E7           <1>       mul    bh
000032D5 0FB7F0         <1>       movzx  esi, ax
000032D8 01EE           <1>       add    esi, ebp
000032DA 88D8           <1>       mov    al, bl
000032DC 28E4           <1>       sub    ah, ah
000032DE 6601D0         <1>       add    ax, dx ; (DX + i)
000032E1 66C1E005       <1>       shl    ax, 5  ; * 32
000032E5 0FB7F8         <1>       movzx  edi, ax
000032E8 81C700000A00   <1>       add    edi, 0A0000h
000032EE 51             <1>       push   ecx
000032EF 0FB6CF         <1>       movzx  ecx, bh
000032F2 F3A4           <1>       rep    movsb
000032F4 59             <1>       pop    ecx
000032F5 FEC3           <1>       inc    bl
000032F7 38CB           <1>       cmp    bl, cl
000032F9 75D6           <1>       jne    short ltup_1
                        <1>       ;
000032FB E840000000     <1>       call   release_font_access
                        <1>       ;
00003300 58             <1>       pop    eax
                        <1>       ; if(AL>=0x10)
00003301 3C10           <1>       cmp    al, 10h
00003303 7205           <1>       jb     short ltup_2
                        <1>       ; set_scan_lines(BH);
00003305 E875000000     <1>       call   set_scan_lines
                        <1> ltup_2:
0000330A 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()
0000330B 52             <1>       push   edx
0000330C 66BAC403       <1>       mov    dx, 3C4h ; VGAREG_SEQU_ADDRESS
00003310 66B80001       <1>       mov    ax, 0100h
00003314 66EF           <1>       out    dx, ax
00003316 66B80204       <1>       mov    ax, 0402h
0000331A 66EF           <1>       out    dx, ax
0000331C 66B80407       <1>       mov    ax, 0704h
00003320 66EF           <1>       out    dx, ax
00003322 66B80003       <1>       mov    ax, 0300h
00003326 66EF           <1>       out    dx, ax
00003328 66BACE03       <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
0000332C 66B80402       <1>       mov    ax, 0204h
00003330 66EF           <1>       out    dx, ax
00003332 66B80500       <1>       mov    ax, 0005h
00003336 66EF           <1>       out    dx, ax
00003338 66B80604       <1>       mov    ax, 0406h
0000333C 66EF           <1>       out    dx, ax
0000333E 5A             <1>       pop    edx
0000333F 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>
00003340 66BAC403       <1>       mov    dx, 3C4h ; VGAREG_SEQU_ADDRESS
00003344 66B80001       <1>       mov    ax, 0100h
00003348 66EF           <1>       out    dx, ax
0000334A 66B80203       <1>       mov    ax, 0302h
0000334E 66EF           <1>       out    dx, ax
00003350 66B80403       <1>       mov    ax, 0304h
00003354 66EF           <1>       out    dx, ax
00003356 66B80003       <1>       mov    ax, 0300h
0000335A 66EF           <1>       out    dx, ax
```

```
6677 0000335C 66BACC03          <1>       mov   dx, 3CCh ; VGAREG_READ_MISC_OUTPUT
6678 00003360 EC                <1>       in    al, dx
6679 00003361 2401              <1>       and   al, 01h
6680 00003363 C0E002            <1>       shl   al, 2
6681 00003366 0C0A              <1>       or    al, 0Ah
6682 00003368 88C4              <1>       mov   ah, al
6683 0000336A B006              <1>       mov   al, 06h
6684 0000336C 66BACE03          <1>       mov   dx, 3CEh ; VGAREG_GRDC_ADDRESS
6685 00003370 66EF              <1>       out   dx, ax
6686 00003372 66B80400          <1>       mov   ax, 0004h
6687 00003376 66EF              <1>       out   dx, ax
6688 00003378 66B80510          <1>       mov   ax, 1005h
6689 0000337C 66EF              <1>       out   dx, ax
6690 0000337E C3                <1>       retn
6691                            <1>
6692                            <1> set_scan_lines:
6693                            <1>       ; 09/07/2016
6694                            <1>       ;
6695                            <1>       ; derived from 'Plex86/Bochs VGABios' source code
6696                            <1>       ; vgabios-0.7a (2011)
6697                            <1>       ; by the LGPL VGABios developers Team (2001-2008)
6698                            <1>       ; 'vgabios.c', 'set_scan_lines'
6699                            <1>
6700                            <1>       ; set_scan_lines(lines)
6701                            <1>       ; BH = lines
6702                            <1>
6703                            <1>       ; outb(crtc_addr, 0x09);
6704 0000337F 66BAD403          <1>       mov   dx, 3D4h ; CRTC_ADDRESS = 3D4h (always)
6705 00003383 B009              <1>       mov   al, 09h
6706 00003385 EE                <1>       out   dx, al
6707                            <1>       ; crtc_r9 = inb(crtc_addr+1);
6708 00003386 6642              <1>       inc   dx ; 3D5h
6709 00003388 EC                <1>       in    al, dx
6710                            <1>       ; crtc_r9 = (crtc_r9 & 0xe0) | (lines - 1);
6711 00003389 24E0              <1>       and   al, 0E0h
6712 0000338B FECF              <1>       dec   bh ; lines - 1
6713 0000338D 08F8              <1>       or    al, bh
6714                            <1>       ; outb(crtc_addr+1, crtc_r9);
6715 0000338F EE                <1>       out   dx, al
6716                            <1>       ;inc  bh
6717                            <1>       ; if(lines==8)
6718                            <1>       ;cmp  bh, 8
6719 00003390 80FF07            <1>       cmp   bh, 7
6720 00003393 7506              <1>       jne   short ssl_1
6721                            <1>       ; biosfn_set_cursor_shape(0x06,0x07);
6722 00003395 66B90706          <1>       mov   cx, 0607h
6723 00003399 EB06              <1>       jmp   short ssl_2
6724                            <1> ssl_1:
6725                            <1>       ; biosfn_set_cursor_shape(lines-4,lines-3);
6726 0000339B 88F9              <1>       mov   cl, bh ; lines - 1
6727 0000339D 88CD              <1>       mov   ch, cl ; lines - 1 (16 -> 15)
6728 0000339F FECD              <1>       dec   ch  ; lines - 2 (16 -> 14)
6729                            <1> ssl_2:
6730                            <1>       ; CH = start line, CL = stop line
6731 000033A1 B40A              <1>       mov   ah, 10 ; 6845 register for cursor set
6732 000033A3 66890D[D36F0000]  <1>       mov   [CURSOR_MODE], cx ; save in data area
6733 000033AA E813F0FFFF        <1>       call  m16   ; output cx register
6734                            <1>       ; write_word(BIOSMEM_SEG,BIOSMEM_CHAR_HEIGHT, lines);
6735 000033AF FEC7              <1>       inc   bh ; lines
6736 000033B1 883D[BE6F0000]    <1>       mov   [CHAR_HEIGHT], bh
6737                            <1>       ;  outb(crtc_addr, 0x12);
6738 000033B7 66BAD403          <1>       mov   dx, 3D4h ; CRTC_ADDRESS
6739 000033BB B012              <1>       mov   al, 12h
6740 000033BD EE                <1>       out   dx, al
6741                            <1>       ; vde = inb(crtc_addr+1);
6742 000033BE 6642              <1>       inc   dx
6743 000033C0 EC                <1>       in    al, dx
6744 000033C1 88C4              <1>       mov   ah, al
6745                            <1>       ; outb(crtc_addr, 0x07);
6746 000033C3 664A              <1>       dec   dx
6747 000033C5 B007              <1>       mov   al, 07h
6748 000033C7 EE                <1>       out   dx, al
6749                            <1>       ; ovl = inb(crtc_addr+1);
6750 000033C8 6642              <1>       inc   dx
6751 000033CA EC                <1>       in    al, dx
6752                            <1>       ; vde += (((ovl & 0x02) << 7) + ((ovl & 0x40) << 3) + 1);
6753 000033CB 88E2              <1>       mov   dl, ah ; vde
6754 000033CD 88C6              <1>       mov   dh, al ; ovl
6755 000033CF 6683E002          <1>       and   ax, 02h
6756 000033D3 66C1E007          <1>       shl   ax, 7
6757 000033D7 6689C1            <1>       mov   cx, ax ; (ovl & 0x02) << 7)
6758 000033DA 88F0              <1>       mov   al, dh ; ovl
6759 000033DC 6683E040          <1>       and   ax, 40h
6760 000033E0 66C1E003          <1>       shl   ax, 3  ; (ovl & 0x40) << 3)
6761 000033E4 6640              <1>       inc   ax ; + 1
6762 000033E6 6601C8            <1>       add   ax, cx
6763 000033E9 30F6              <1>       xor   dh, dh
6764 000033EB 6601D0            <1>       add   ax, dx ; + vde
6765                            <1>       ; rows = vde / lines;
6766 000033EE F6F7              <1>       div   bh
6767                            <1>       ;dec  al ; rows -1
6768                            <1>       ; write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, rows-1);
6769 000033F0 A2[C26F0000]      <1>       mov   [VGA_ROWS], al ; rows (not 'rows-1' !)
6770                            <1>       ; write_word(BIOSMEM_SEG,BIOSMEM_PAGE_SIZE, rows * cols * 2);
6771                            <1>       ;mov  ah, [CRT_COLS]
6772                            <1>       ;mul  ah
6773                            <1>       ; 17/11/2020
6774 000033F5 F625[BC6F0000]    <1>       mul   byte [CRT_COLS]
6775 000033FB 66D1E0            <1>       shl   ax, 1
6776 000033FE 66A3[288E0100]    <1>       mov   [CRT_LEN], ax
6777 00003404 C3                <1>       retn
6778                            <1>
6779                            <1> load_text_8_14_pat:
6780                            <1>       ; 26/07/2016
6781                            <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>
00003405 50                     <1>       push   eax
00003406 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>
0000340B 28DB                   <1>       sub    bl, bl ; i = 0
0000340D B70E                   <1>       mov    bh, 14
0000340F BE[24600100]           <1>       mov    esi, vgafont14
00003414 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
00003419 0FB6CF                 <1>       movzx  ecx, bh
0000341C F3A4                   <1>       rep    movsb
0000341E 83C712                 <1>       add    edi, 18 ; 32 - 14
00003421 FEC3                   <1>       inc    bl
00003423 75F4                   <1>       jnz    short lt8_14_1
                                <1>       ;
00003425 E816FFFFFF             <1>       call   release_font_access
                                <1>       ;
0000342A 58                     <1>       pop    eax
                                <1>       ; if(AL>=0x10)
0000342B 3C10                   <1>       cmp    al, 10h
0000342D 7205                   <1>       jb     short lt8_14_4
                                <1>       ; BH = 14
                                <1>       ; set_scan_lines(14);
0000342F E84BFFFFFF             <1>       call   set_scan_lines
                                <1> lt8_14_4:
00003434 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>
00003435 50                     <1>       push   eax
00003436 E8D0FEFFFF             <1>       call   get_font_access
                                <1>
```

```
6887                                    <1>       ; blockaddr = ((BL & 0x03) << 14) + ((BL & 0x04) << 11);
6888                                    <1>       ;mov   dl, bl
6889                                    <1>       ;and   dl, 3
6890                                    <1>       ;shl   dx, 14
6891                                    <1>       ;xchg  dx, bx
6892                                    <1>       ;and   dl, 4
6893                                    <1>       ;shl   dx, 11
6894                                    <1>       ;add   dx, bx
6895                                    <1>
6896                                    <1>       ;xor   dx, dx  ; blockaddr = 0
6897                                    <1>       ; Always block 0 for TRDOS 386 ! (blockaddr=0)
6898                                    <1>
6899 0000343B 28DB                     <1>       sub    bl, bl ; i = 0
6900 0000343D B708                     <1>       mov    bh, 8
6901                                    <1>       ;mov   esi, vgafont8
6902 0000343F BF00000A00               <1>       mov    edi, 0A0000h
6903                                    <1>
6904                                    <1>       ; 05/01/2021
6905 00003444 F605[82120300]80         <1>       test   byte [ufont], 80h
6906 0000344B 7410                     <1>       jz     short lt8_8_0
6907                                    <1>              ; user font permission (after set mode)
6908 0000344D F605[82120300]08         <1>       test   byte [ufont], 8
6909 00003454 7407                     <1>       jz     short lt8_8_0
6910 00003456 BE00500900               <1>       mov    esi, VGAFONT8USER
6911 0000345B EB05                     <1>       jmp    short lt8_8_1
6912                                    <1> lt8_8_0:
6913 0000345D BE[24580100]             <1>       mov    esi, vgafont8
6914                                    <1> lt8_8_1:
6915                                    <1>       ;mov   al, bl
6916                                    <1>       ;mul   bh
6917                                    <1>       ;movzx esi, ax
6918                                    <1>       ;add   esi, vgafont8
6919                                    <1>       ;mov   al, bl
6920                                    <1>       ;sub   ah, ah
6921                                    <1>       ;shl   ax, 5 ; * 32
6922                                    <1>       ;;add  ax, dx ; blockaddr + i * 32;
6923                                    <1>       ;movzx edi, ax ; dest
6924                                    <1>       ;add   edi, 0A0000h
6925 00003462 0FB6CF                   <1>       movzx  ecx, bh
6926 00003465 F3A4                     <1>       rep    movsb
6927 00003467 83C718                   <1>       add    edi, 24 ; 32 - 8
6928 0000346A FEC3                     <1>       inc    bl
6929 0000346C 75F4                     <1>       jnz    short lt8_8_1
6930                                    <1>       ;
6931 0000346E E8CDFEFFFF               <1>       call   release_font_access
6932                                    <1>       ;
6933 00003473 58                       <1>       pop    eax
6934                                    <1>       ; if(AL>=0x10)
6935 00003474 3C10                     <1>       cmp    al, 10h
6936 00003476 7205                     <1>       jb     short lt8_8_2
6937                                    <1>       ; BH = 8
6938                                    <1>       ; set_scan_lines(8);
6939 00003478 E802FFFFFF               <1>       call   set_scan_lines
6940                                    <1> lt8_8_2:
6941 0000347D C3                       <1>       retn
6942                                    <1>
6943                                    <1> load_text_8_16_pat:
6944                                    <1>       ; 05/01/2021 (TRDOS 386 v2.0.3)
6945                                    <1>       ; 26/07/2016
6946                                    <1>       ; 25/07/2016
6947                                    <1>       ; 23/07/2016
6948                                    <1>       ; 09/07/2016
6949                                    <1>       ; load user defined (EGA/VGA) text fonts
6950                                    <1>       ;
6951                                    <1>       ; derived from 'Plex86/Bochs VGABios' source code
6952                                    <1>       ; vgabios-0.7a (2011)
6953                                    <1>       ; by the LGPL VGABios developers Team (2001-2008)
6954                                    <1>       ; 'vgabios.c', 'biosfn_load_text_8_16_pat'
6955                                    <1>
6956                                    <1>       ; biosfn_load_text_8_16_pat (AL,BL)
6957                                    <1>
6958                                    <1>       ; get_font_access();
6959                                    <1>       ; blockaddr = ((BL & 0x03) << 14) + ((BL & 0x04) << 11);
6960                                    <1>       ; for(i=0;i<0x100;i++)
6961                                    <1>       ; {
6962                                    <1>       ;  src = i * 16;
6963                                    <1>       ;  dest = blockaddr + i * 32;
6964                                    <1>       ;  memcpyb(0xA000, dest, 0xC000, vgafont16+src, 16);
6965                                    <1>       ; }
6966                                    <1>       ; release_font_access();
6967                                    <1>       ; if(AL>=0x10)
6968                                    <1>       ; {
6969                                    <1>       ; set_scan_lines(16);
6970                                    <1>       ; }
6971                                    <1>
6972 0000347E 50                       <1>       push   eax
6973 0000347F E887FEFFFF               <1>       call   get_font_access
6974                                    <1>
6975                                    <1>       ; blockaddr = ((BL & 0x03) << 14) + ((BL & 0x04) << 11);
6976                                    <1>       ;mov   dl, bl
6977                                    <1>       ;and   dl, 3
6978                                    <1>       ;shl   dx, 14
6979                                    <1>       ;xchg  dx, bx
6980                                    <1>       ;and   dl, 4
6981                                    <1>       ;shl   dx, 11
6982                                    <1>       ;add   dx, bx
6983                                    <1>
6984                                    <1>       ;xor   dx, dx  ; blockaddr = 0
6985                                    <1>       ; Always block 0 for TRDOS 386 ! (blockaddr=0)
6986                                    <1>
6987 00003484 28DB                     <1>       sub    bl, bl ; i = 0
6988 00003486 B710                     <1>       mov    bh, 16
6989                                    <1>       ;mov   esi, vgafont16
6990 00003488 BF00000A00               <1>       mov    edi, 0A0000h
6991 0000348D 0FB6C7                   <1>       movzx  eax, bh
```

```
                                <1>
                                <1>       ; 05/01/2021
00003490 F605[82120300]80       <1>       test  byte [ufont], 80h
00003497 7410                   <1>       jz    short lt8_16_0
                                <1>             ; user font permission (after set mode)
00003499 F605[82120300]10       <1>       test  byte [ufont], 16
000034A0 7407                   <1>       jz    short lt8_16_0
000034A2 BE00400900             <1>       mov   esi, VGAFONT16USER
000034A7 EB05                   <1>       jmp   short lt8_16_1
                                <1> lt8_16_0:
000034A9 BE[246E0100]           <1>       mov   esi, vgafont16
                                <1> lt8_16_1:
                                <1>       ;mov  al, bl
                                <1>       ;mul  bh
                                <1>       ;movzx esi, ax
                                <1>       ;add  esi, vgafont16
                                <1>       ;mov  al, bl ; i
                                <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
                                <1>       ;movzx ecx, bh
000034AE 89C1                   <1>       mov   ecx, eax ; 16
000034B0 F3A4                   <1>       rep   movsb
000034B2 01C7                   <1>       add   edi, eax ; add edi, 16
000034B4 FEC3                   <1>       inc   bl
000034B6 75F6                   <1>       jnz   short lt8_16_1
                                <1>       ;
000034B8 E883FEFFFF             <1>       call  release_font_access
                                <1>       ;
000034BD 58                     <1>       pop   eax
                                <1>       ; if(AL>=0x10)
000034BE 3C10                   <1>       cmp   al, 10h
000034C0 7205                   <1>       jb    short lt8_16_2
                                <1>       ; BH = 16
                                <1>       ; set_scan_lines(16);
000034C2 E8B8FEFFFF             <1>       call  set_scan_lines
                                <1> lt8_16_2:
000034C7 C3                     <1>       retn
                                <1>
                                <1> load_gfx_user_chars:
                                <1>       ; 08/01/2021
                                <1>       ; 05/01/2021 (TRDOS 386 v2.0.3)
                                <1>       ; 08/08/2016
                                <1>       ; 10/07/2016
                                <1>       ; Setup User-Defined 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_user_chars'
                                <1>
                                <1>       ; biosfn_load_gfx_user_chars (ES,BP,CX,BL,DL)
                                <1>       ; /* set 0x43 INT pointer */
                                <1>       ; write_word(0x0, 0x43*4, BP);
                                <1>       ; write_word(0x0, 0x43*4+2, ES);
                                <1>
                                <1>       ; 08/01/2021
                                <1>
                                <1>       ; BL   screen rows code: 00H = user-specified (in DL)
                                <1>         ;                      01H = 14 rows
                                <1>         ;                      02H = 25 rows
                                <1>         ;                      03H = 43 rows
                                <1>         ; CX   bytes per character definition
                                <1>         ; DL   (when BL=0) custom number of character rows on screen
                                <1>         ; EBP  address of font-definition information (user's mem space)
                                <1>
                                <1>       ; 05/01/2021
                                <1>       ;xor   eax, eax
                                <1>       ;dec   eax ; 0FFFFFFFFh (user defined fonts)
                                <1>       ;mov   [VGA_INT43H], eax
                                <1>
                                <1>       ; 08/01/2021
                                <1>       ; ebp = video font data (buffer) address
000034C8 892D[3A8E0100]         <1>       mov   [VGA_INT43H], ebp
                                <1>
                                <1>       ; switch (BL) {
                                <1>       ; case 0:
                                <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, DL-1);
                                <1>       ;    break;
000034CE 20DB                   <1>       and   bl, bl
000034D0 7508                   <1>       jnz   short l_gfx_uc_1
000034D2 8815[C26F0000]         <1>       mov   [VGA_ROWS], dl  ; not DL-1 !
000034D8 EB23                   <1>       jmp   short l_gfx_uc_4
                                <1> l_gfx_uc_1:
                                <1>       ; case 1:
                                <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 13);
                                <1>       ;    break;
000034DA FECB                   <1>       dec   bl
000034DC 7509                   <1>       jnz   short l_gfx_uc_2
                                <1>       ; bl = 1
000034DE C605[C26F0000]0E       <1>       mov   byte [VGA_ROWS], 14  ; not 13 !
000034E5 EB16                   <1>       jmp   short l_gfx_uc_4
                                <1> l_gfx_uc_2:
000034E7 FECB                   <1>       dec   bl
000034E9 740B                   <1>       jz    short l_gfx_uc_3 ; bl = 2
000034EB FECB                   <1>       dec   bl
000034ED 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;
000034EF C605[C26F0000]2B       <1>       mov   byte [VGA_ROWS], 43  ; not 42 !
                                <1> l_gfx_uc_3:
```

```
7097                                    <1>       ; case 2:
7098                                    <1>       ; default:
7099                                    <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 24);
7100                                    <1>       ;    break;
7101                                    <1>       ; bl = 2 or bl > 3
7102 000034F6 C605[C26F0000]19         <1>       mov    byte [VGA_ROWS], 25  ; not 24 !
7103                                    <1>       ; }
7104                                    <1> l_gfx_uc_4:
7105                                    <1>       ; write_byte(BIOSMEM_SEG, BIOSMEM_CHAR_HEIGHT, CX);
7106 000034FD 880D[BE6F0000]            <1>       mov    [CHAR_HEIGHT], cl
7107                                    <1>       ; }
7108 00003503 C3                       <1>       retn
7109                                    <1>
7110                                    <1> load_gfx_8_14_chars:
7111                                    <1>       ; 08/08/2016
7112                                    <1>       ; 10/07/2016
7113                                    <1>       ; Setup ROM 8x14 Font for Graphics Mode (VGA)
7114                                    <1>       ;
7115                                    <1>       ; derived from 'Plex86/Bochs VGABios' source code
7116                                    <1>       ; vgabios-0.7a (2011)
7117                                    <1>       ; by the LGPL VGABios developers Team (2001-2008)
7118                                    <1>       ; 'vgabios.c', 'biosfn_load_gfx_8_14_chars'
7119                                    <1>
7120                                    <1>       ; biosfn_load_gfx_8_14_chars (BL)
7121                                    <1>       ; /* set 0x43 INT pointer */
7122                                    <1>       ; write_word(0x0, 0x43*4, &vgafont14);
7123                                    <1>       ; write_word(0x0, 0x43*4+2, 0xC000);
7124 00003504 C705[3A8E0100]-          <1>       mov    dword [VGA_INT43H], vgafont14
7124 0000350A [24600100]               <1>
7125                                    <1>
7126                                    <1>       ; BL    screen rows code: 00H = user-specified (in DL)
7127                                    <1>       ;                         01H = 14 rows
7128                                    <1>       ;                         02H = 25 rows
7129                                    <1>       ;                         03H = 43 rows
7130                                    <1>       ; DL    (when BL=0) custom number of char rows on screen
7131                                    <1>
7132                                    <1>       ; switch (BL) {
7133                                    <1>       ; case 0:
7134                                    <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, DL-1);
7135                                    <1>       ;    break;
7136 0000350E 20DB                     <1>       and    bl, bl
7137 00003510 7508                     <1>       jnz    short l_gfx_8_14c_1
7138 00003512 8815[C26F0000]           <1>       mov    [VGA_ROWS], dl  ; not DL-1 !
7139 00003518 EB23                     <1>       jmp    short l_gfx_8_14c_4
7140                                    <1> l_gfx_8_14c_1:
7141                                    <1>       ; case 1:
7142                                    <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 13);
7143                                    <1>       ;    break;
7144 0000351A FECB                     <1>       dec    bl
7145 0000351C 7509                     <1>       jnz    short l_gfx_8_14c_2
7146                                    <1>       ; bl = 1
7147 0000351E C605[C26F0000]0E         <1>       mov    byte [VGA_ROWS], 14  ; not 13 !
7148 00003525 EB16                     <1>       jmp    short l_gfx_8_14c_4
7149                                    <1> l_gfx_8_14c_2:
7150 00003527 FECB                     <1>       dec    bl
7151 00003529 740B                     <1>       jz     short l_gfx_8_14c_3 ; bl = 2
7152 0000352B FECB                     <1>       dec    bl
7153 0000352D 750E                     <1>       jnz    short l_gfx_8_14c_4 ; bl > 3
7154                                    <1>       ; bl = 3
7155                                    <1>       ; case 3:
7156                                    <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 42);
7157                                    <1>       ;    break;
7158 0000352F C605[C26F0000]2B         <1>       mov    byte [VGA_ROWS], 43  ; not 42 !
7159                                    <1> l_gfx_8_14c_3:
7160                                    <1>       ; case 2:
7161                                    <1>       ; default:
7162                                    <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 24);
7163                                    <1>       ;    break;
7164                                    <1>       ; bl = 2 or bl > 3
7165 00003536 C605[C26F0000]19         <1>       mov    byte [VGA_ROWS], 25  ; not 24 !
7166                                    <1>       ; }
7167                                    <1> l_gfx_8_14c_4:
7168                                    <1>       ; write_byte(BIOSMEM_SEG, BIOSMEM_CHAR_HEIGHT, 14);
7169 0000353D C605[BE6F0000]0E         <1>         mov     byte [CHAR_HEIGHT], 14
7170                                    <1>       ; }
7171 00003544 C3                       <1>       retn
7172                                    <1>
7173                                    <1> load_gfx_8_8_chars:
7174                                    <1>       ; 08/08/2016
7175                                    <1>       ; 10/07/2016
7176                                    <1>       ; Setup ROM 8x14 Font for Graphics Mode (VGA)
7177                                    <1>       ;
7178                                    <1>       ; derived from 'Plex86/Bochs VGABios' source code
7179                                    <1>       ; vgabios-0.7a (2011)
7180                                    <1>       ; by the LGPL VGABios developers Team (2001-2008)
7181                                    <1>       ; 'vgabios.c', 'biosfn_load_gfx_8_8_dd_chars'
7182                                    <1>
7183                                    <1>       ; biosfn_load_gfx_8_8_dd_chars (BL)
7184                                    <1>       ; /* set 0x43 INT pointer */
7185                                    <1>       ; write_word(0x0, 0x43*4, &vgafont8);
7186                                    <1>       ; write_word(0x0, 0x43*4+2, 0xC000);
7187 00003545 C705[3A8E0100]-          <1>       mov    dword [VGA_INT43H], vgafont8
7187 0000354B [24580100]               <1>
7188                                    <1>
7189                                    <1>       ; BL    screen rows code: 00H = user-specified (in DL)
7190                                    <1>       ;                         01H = 14 rows
7191                                    <1>       ;                         02H = 25 rows
7192                                    <1>       ;                         03H = 43 rows
7193                                    <1>       ; DL    (when BL=0) custom number of char rows on screen
7194                                    <1>
7195                                    <1>       ; switch (BL) {
7196                                    <1>       ; case 0:
7197                                    <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, DL-1);
7198                                    <1>       ;    break;
7199 0000354F 20DB                     <1>       and    bl, bl
```

```
7200 00003551 7508              <1>       jnz    short l_gfx_8_8c_1
7201 00003553 8815[C26F0000]    <1>       mov    [VGA_ROWS], dl  ; not DL-1 !
7202 00003559 EB23              <1>       jmp    short l_gfx_8_8c_4
7203                            <1> l_gfx_8_8c_1:
7204                            <1>       ; case 1:
7205                            <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 13);
7206                            <1>       ;    break;
7207 0000355B FECB              <1>       dec    bl
7208 0000355D 7509              <1>       jnz    short l_gfx_8_8c_2
7209                            <1>       ; bl = 1
7210 0000355F C605[C26F0000]0E  <1>       mov    byte [VGA_ROWS], 14  ; not 13 !
7211 00003566 EB16              <1>       jmp    short l_gfx_8_8c_4
7212                            <1> l_gfx_8_8c_2:
7213 00003568 FECB              <1>       dec    bl
7214 0000356A 740B              <1>       jz     short l_gfx_8_8c_3 ; bl = 2
7215 0000356C FECB              <1>       dec    bl
7216 0000356E 750E              <1>       jnz    short l_gfx_8_8c_4 ; bl > 3
7217                            <1>       ; bl = 3
7218                            <1>       ; case 3:
7219                            <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 42);
7220                            <1>       ;    break;
7221 00003570 C605[C26F0000]2B  <1>       mov    byte [VGA_ROWS], 43  ; not 42 !
7222                            <1> l_gfx_8_8c_3:
7223                            <1>       ; case 2:
7224                            <1>       ; default:
7225                            <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 24);
7226                            <1>       ;    break;
7227                            <1>       ; bl = 2 or bl > 3
7228 00003577 C605[C26F0000]19  <1>       mov    byte [VGA_ROWS], 25  ; not 24 !
7229                            <1>       ; }
7230                            <1> l_gfx_8_8c_4:
7231                            <1>       ; write_byte(BIOSMEM_SEG, BIOSMEM_CHAR_HEIGHT, 8);
7232 0000357E C605[BE6F0000]08  <1>         mov     byte [CHAR_HEIGHT], 8
7233                            <1>       ; }
7234 00003585 C3                <1>       retn
7235                            <1>
7236                            <1> load_gfx_8_16_chars:
7237                            <1>       ; 08/08/2016
7238                            <1>       ; 10/07/2016
7239                            <1>       ; Setup ROM 8x14 Font for Graphics Mode (VGA)
7240                            <1>       ;
7241                            <1>       ; derived from 'Plex86/Bochs VGABios' source code
7242                            <1>       ; vgabios-0.7a (2011)
7243                            <1>       ; by the LGPL VGABios developers Team (2001-2008)
7244                            <1>       ; 'vgabios.c', 'biosfn_load_gfx_8_16_chars'
7245                            <1>
7246                            <1>       ; biosfn_load_gfx_8_16_chars (BL)
7247                            <1>       ; /* set 0x43 INT pointer */
7248                            <1>       ; write_word(0x0, 0x43*4, &vgafont16);
7249                            <1>       ; write_word(0x0, 0x43*4+2, 0xC000);
7250 00003586 C705[3A8E0100]-   <1>       mov    dword [VGA_INT43H], vgafont16
7250 0000358C [246E0100]        <1>
7251                            <1>
7252                            <1>       ; BL    screen rows code: 00H = user-specified (in DL)
7253                            <1>         ;                       01H = 14 rows
7254                            <1>         ;                       02H = 25 rows
7255                            <1>         ;                       03H = 43 rows
7256                            <1>         ; DL    (when BL=0) custom number of char rows on screen
7257                            <1>
7258                            <1>       ; switch (BL) {
7259                            <1>       ; case 0:
7260                            <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, DL-1);
7261                            <1>       ;    break;
7262 00003590 20DB              <1>       and    bl, bl
7263 00003592 7508              <1>       jnz    short l_gfx_8_16c_1
7264 00003594 8815[C26F0000]    <1>       mov    [VGA_ROWS], dl  ; not DL-1 !
7265 0000359A EB23              <1>       jmp    short l_gfx_8_16c_4
7266                            <1> l_gfx_8_16c_1:
7267                            <1>       ; case 1:
7268                            <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 13);
7269                            <1>       ;    break;
7270 0000359C FECB              <1>       dec    bl
7271 0000359E 7509              <1>       jnz    short l_gfx_8_16c_2
7272                            <1>       ; bl = 1
7273 000035A0 C605[C26F0000]0E  <1>       mov    byte [VGA_ROWS], 14  ; not 13 !
7274 000035A7 EB16              <1>       jmp    short l_gfx_8_16c_4
7275                            <1> l_gfx_8_16c_2:
7276 000035A9 FECB              <1>       dec    bl
7277 000035AB 740B              <1>       jz     short l_gfx_8_16c_3 ; bl = 2
7278 000035AD FECB              <1>       dec    bl
7279 000035AF 750E              <1>       jnz    short l_gfx_8_16c_4 ; bl > 3
7280                            <1>       ; bl = 3
7281                            <1>       ; case 3:
7282                            <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 42);
7283                            <1>       ;    break;
7284 000035B1 C605[C26F0000]2B  <1>       mov    byte [VGA_ROWS], 43  ; not 42 !
7285                            <1> l_gfx_8_16c_3:
7286                            <1>       ; case 2:
7287                            <1>       ; default:
7288                            <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 24);
7289                            <1>       ;    break;
7290                            <1>       ; bl = 2 or bl > 3
7291 000035B8 C605[C26F0000]19  <1>       mov    byte [VGA_ROWS], 25  ; not 24 !
7292                            <1>       ; }
7293                            <1> l_gfx_8_16c_4:
7294                            <1>       ; write_byte(BIOSMEM_SEG, BIOSMEM_CHAR_HEIGHT, 16);
7295 000035BF C605[BE6F0000]10  <1>         mov     byte [CHAR_HEIGHT], 16
7296                            <1>       ; }
7297 000035C6 C3                <1>       retn
7298                            <1>
7299                            <1> get_font_info:
7300                            <1>       ; 08/01/2021 (TRDOS 386 v2.0.3)
7301                            <1>       ; 19/09/2016
7302                            <1>       ; 08/08/2016
7303                            <1>       ; 10/07/2016
```

```
7304                                 <1>       ; Get Current Character Generator Info (VGA)
7305                                 <1>       ;
7306                                 <1>       ; derived from 'Plex86/Bochs VGABios' source code
7307                                 <1>       ; vgabios-0.7a (2011)
7308                                 <1>       ; by the LGPL VGABios developers Team (2001-2008)
7309                                 <1>       ; 'vgabios.c', 'biosfn_get_font_info'
7310                                 <1>
7311                                 <1>       ; Modified for TRDOS 386 !
7312                                 <1>       ;
7313                                 <1>       ; INPUT ->
7314                                 <1>       ;    AX = 1130h
7315                                 <1>       ;    BL = 0 -> Get info for current VGA font
7316                                 <1>       ;              (BH = unused)
7317                                 <1>       ;    19/09/2016
7318                                 <1>       ;    BL > 0 -> Get requested character font data
7319                                 <1>       ;       BL = 1 -> vgafont8
7320                                 <1>       ;         BL = 2 -> vgafont14
7321                                 <1>       ;       BL = 3 -> vgafont16
7322                                 <1>       ;      ;08/01/2021
7323                                 <1>       ;       BL = 4 -> user defined 8x8 font
7324                                 <1>       ;       BL = 5 -> user defined 8x14 font
7325                                 <1>       ;       BL = 6 -> user defined 8x16 font
7326                                 <1>       ;       BL > 6 -> Invalid function (for now!)
7327                                 <1>       ;       BH = ASCII code of the first character
7328                                 <1>       ;       ECX = Number of characters from the 1st char
7329                                 <1>       ;       ECX >= 256 -> All (256-BH) characters
7330                                 <1>       ;       ECX = 0 -> All characters (BH = unused)
7331                                 <1>       ;         EDX = User's Buffer Address
7332                                 <1>       ; OUTPUT ->
7333                                 <1>       ;    AL = height (scanlines), bytes per character
7334                                 <1>       ;    AH = screen rows
7335                                 <1>       ;    Byte 16-23 of EAX = number of columns
7336                                 <1>       ;    Byte 24-31 of EAX =
7337                                 <1>       ;         0 -> default font (not configured yet)
7338                                 <1>       ;         0FFh -> user defined font
7339                                 <1>       ;         14 = vgafont14
7340                                 <1>       ;          8 = vgafont8
7341                                 <1>       ;         16 = vgafont16
7342                                 <1>       ;   If BL input > 0 ->
7343                                 <1>       ;        EAX = Actual transfer count
7344                                 <1>       ;
7345 000035C7 20DB                  <1>       and    bl, bl
7346 000035C9 740F                  <1>       jz     short gfi_1
7347                                 <1>       ; invalid function (input)
7348                                 <1>       ; 08/01/2021
7349 000035CB 80FB04                <1>       cmp    bl, 4
7350 000035CE 7263                  <1>       jb     short gfi_5
7351 000035D0 7441                  <1>       je     short gfi_3
7352 000035D2 80FB06                <1>       cmp    bl, 6
7353 000035D5 744C                  <1>       je     short gfi_4
7354                                 <1>       ; bh = 5 or bh > 6
7355                                 <1> gfi_0:
7356 000035D7 31C0                  <1>       xor    eax, eax ; 0
7357 000035D9 C3                    <1>       retn
7358                                 <1> gfi_1:
7359 000035DA A0[BE6F0000]          <1>       mov    al, [CHAR_HEIGHT]
7360 000035DF 8A25[C26F0000]        <1>       mov    ah, [VGA_ROWS]
7361 000035E5 C1E010                <1>       shl    eax, 16
7362 000035E8 A0[BC6F0000]          <1>       mov    al, [CRT_COLS]
7363 000035ED 8B0D[3A8E0100]        <1>       mov    ecx, [VGA_INT43H]
7364 000035F3 21C9                  <1>       and    ecx, ecx
7365 000035F5 7418                  <1>       jz     short gfi_2 ; 0 = default font
7366                                 <1>       ; 08/01/2021
7367 000035F7 FECC                  <1>       dec    ah ; 0FFh
7368 000035F9 81F900400900          <1>       cmp    ecx, VGAFONT16USER
7369 000035FF 740E                  <1>       je     short gfi_2
7370 00003601 81F900500900          <1>       cmp    ecx, VGAFONT8USER
7371 00003607 7406                  <1>       je     short gfi_2
7372 00003609 8A25[BE6F0000]        <1>       mov    ah, [CHAR_HEIGHT] ; font size = height
7373                                 <1> gfi_2:
7374 0000360F C1C010                <1>       rol    eax, 16
7375 00003612 C3                    <1>       retn
7376                                 <1> gfi_3:
7377                                 <1>       ; 08/01/2021
7378 00003613 F605[82120300]08      <1>       test   byte [ufont], 08h ; 8x8 user font
7379 0000361A 74BB                  <1>       jz     short gfi_0 ; not loaded !
7380 0000361C BE00500900            <1>       mov    esi, VGAFONT8USER ; *
7381                                 <1>       ;mov   bl, 8
7382                                 <1>       ;jmp   short gfi_8
7383 00003621 EB4D                  <1>       jmp    short gfi_10
7384                                 <1> gfi_4:
7385                                 <1>       ; 08/01/2021
7386 00003623 F605[82120300]10      <1>       test   byte [ufont], 10h ; 8x16 user font
7387 0000362A 74AB                  <1>       jz     short gfi_0 ; not loaded !
7388 0000362C BE00400900            <1>       mov    esi, VGAFONT16USER ; *
7389 00003631 EB15                  <1>       jmp    short gfi_7
7390                                 <1> gfi_5:
7391 00003633 80FB02                <1>       cmp    bl, 2
7392 00003636 7233                  <1>       jb     short gfi_9
7393 00003638 7709                  <1>       ja     short gfi_6
7394                                 <1>       ;BL = 2 -> vgafont14
7395 0000363A BE[24600100]          <1>       mov    esi, vgafont14 ; *
7396 0000363F B30E                  <1>       mov    bl, 14
7397 00003641 EB07                  <1>       jmp    short gfi_8
7398                                 <1> gfi_6:
7399                                 <1>       ;BL = 3 -> vgafont16
7400 00003643 BE[246E0100]          <1>       mov    esi, vgafont16 ; *
7401                                 <1> gfi_7:
7402 00003648 B310                  <1>       mov    bl, 16
7403                                 <1> gfi_8:
7404 0000364A 89D7                  <1>       mov    edi, edx ; **
7405 0000364C 09C9                  <1>       or     ecx, ecx
7406 0000364E 7424                  <1>       jz     short gfi_11 ; all chars from the 00h
7407 00003650 88F8                  <1>       mov    al, bh ; character index
7408 00003652 F6E3                  <1>       mul    bl ; char index * char height/size
```

```
7409 00003654 0FB7D0            <1>       movzx  edx, ax
7410 00003657 01D6              <1>       add    esi, edx ; *
7411 00003659 66BAFF00          <1>       mov    dx, 255
7412 0000365D 28FA              <1>       sub    dl, bh
7413 0000365F 6642              <1>       inc    dx
7414 00003661 39D1              <1>       cmp    ecx, edx
7415 00003663 770F              <1>       ja     short gfi_11
7416 00003665 7412              <1>       je     short gfi_12
7417 00003667 89D1              <1>       mov    ecx, edx
7418 00003669 EB0E              <1>       jmp    short gfi_12
7419                            <1> gfi_9:
7420                            <1>       ;BL = 1 -> vgafont8
7421 0000366B BE[24580100]      <1>       mov    esi, vgafont8 ; *
7422                            <1> gfi_10:
7423 00003670 B308              <1>       mov    bl, 8
7424 00003672 EBD6              <1>       jmp    short gfi_8
7425                            <1> gfi_11:
7426 00003674 B900010000        <1>       mov    ecx, 256
7427                            <1> gfi_12:
7428                            <1>       ; 08/01/2021
7429 00003679 89C8              <1>       mov    eax, ecx ; character count
7430 0000367B 30FF              <1>       xor    bh, bh
7431 0000367D 66F7E3            <1>       mul    bx ; char count * char height/size
7432 00003680 89C1              <1>       mov    ecx, eax
7433                            <1>
7434                            <1>       ; ESI = source address in system space
7435                            <1>       ; EDI = user's buffer address
7436                            <1>       ; ECX = transfer (byte) count
7437 00003682 E83DDC0000        <1>       call   transfer_to_user_buffer
7438 00003687 89C8              <1>       mov    eax, ecx ; actual transfer count
7439 00003689 C3                <1>       retn
7440                            <1>
7441                            <1> vga_pal_funcs:
7442                            <1>       ; 10/08/2016
7443                            <1>       ; VGA Palette functions
7444                            <1>       ;
7445                            <1>       ; derived from 'Plex86/Bochs VGABios' source code
7446                            <1>       ; vgabios-0.7a (2011)
7447                            <1>       ; by the LGPL VGABios developers Team (2001-2008)
7448                            <1>       ; 'vgabios.c', 'vgarom.asm'
7449                            <1>
7450 0000368A 3C00              <1>       cmp    al, 0
7451 0000368C 0F848F000000      <1>         je      set_single_palette_reg
7452                            <1> vga_palf_1001:
7453 00003692 3C01              <1>       cmp    al, 1
7454 00003694 0F84B4000000      <1>         je      set_overscan_border_color
7455                            <1> vga_palf_1002:
7456 0000369A 3C02              <1>       cmp    al, 2
7457 0000369C 0F84B0000000      <1>         je      set_all_palette_reg
7458                            <1> vga_palf_1003:
7459 000036A2 3C03              <1>       cmp    al, 3
7460 000036A4 0F84E8000000      <1>         je      toggle_intensity
7461                            <1> vga_palf_1007:
7462 000036AA 3C07              <1>       cmp    al, 7
7463 000036AC 0F840D010000      <1>         je      get_single_palette_reg
7464 000036B2 7266              <1>       jb     short vga_palf_unknown
7465                            <1> vga_palf_1008:
7466 000036B4 3C08              <1>       cmp    al, 8
7467 000036B6 0F8437010000      <1>         je      read_overscan_border_color
7468                            <1> vga_palf_1009:
7469 000036BC 3C09              <1>       cmp    al, 9
7470 000036BE 0F8433010000      <1>       je     get_all_palette_reg
7471                            <1> vga_palf_1010:
7472 000036C4 3C10              <1>       cmp    al, 10h
7473 000036C6 0F8487010000      <1>       je     set_single_dac_reg
7474 000036CC 724C              <1>       jb     short vga_palf_unknown
7475                            <1> vga_palf_1012:
7476 000036CE 3C12              <1>       cmp    al, 12h
7477 000036D0 0F8498010000      <1>         je      set_all_dac_reg
7478 000036D6 7242              <1>       jb     short vga_palf_unknown
7479                            <1> vga_palf_1013:
7480 000036D8 3C13              <1>       cmp    al, 13h
7481 000036DA 0F84CC010000      <1>         je      select_video_dac_color_page
7482                            <1> vga_palf_1015:
7483 000036E0 3C15              <1>       cmp    al, 15h
7484 000036E2 0F8412020000      <1>       je     read_single_dac_reg
7485 000036E8 7230              <1>       jb     short vga_palf_unknown
7486                            <1> vga_palf_1017:
7487 000036EA 3C17              <1>       cmp    al, 17h
7488 000036EC 0F8428020000      <1>         je      read_all_dac_reg
7489 000036F2 7226              <1>       jb     short vga_palf_unknown
7490                            <1> vga_palf_1018:
7491 000036F4 3C18              <1>       cmp    al, 18h
7492 000036F6 0F845E020000      <1>         je      set_pel_mask
7493                            <1> vga_palf_1019:
7494 000036FC 3C19              <1>       cmp    al, 19h
7495 000036FE 0F8462020000      <1>         je      read_pel_mask
7496                            <1> vga_palf_101A:
7497 00003704 3C1A              <1>       cmp    al, 1Ah
7498 00003706 0F8468020000      <1>         je      read_video_dac_state
7499                            <1> vga_palf_101B:
7500 0000370C 3C1B              <1>       cmp    al, 1Bh
7501                            <1>       ;jne   short vga_palf_unknown
7502 0000370E 770A              <1>       ja     short vga_palf_unknown
7503                            <1>
7504 00003710 E8C3F5FFFF        <1>       call   gray_scale_summing
7505 00003715 E93DE4FFFF        <1>         jmp     VIDEO_RETURN
7506                            <1>
7507                            <1> vga_palf_unknown:
7508 0000371A 29C0              <1>       sub    eax, eax ; 0 = invalid function
7509 0000371C E93BE4FFFF        <1>         jmp     _video_return
7510                            <1>
7511                            <1> set_single_palette_reg:
7512                            <1>       ; 10/08/2016
7513                            <1>       ; Set One Palette Register
```

```
7514                            <1>       ; BL = register number to set
7515                            <1>       ;      (a 4-bit attribute nibble: 00h-0Fh)
7516                            <1>       ; BH = 6-bit RGB color to display
7517                            <1>       ;      for that attribute
7518                            <1>
7519 00003721 80FB14           <1>       cmp    bl, 14h
7520                            <1>       ;ja    short no_actl_reg1
7521 00003724 0F872DE4FFFF      <1>       ja     VIDEO_RETURN
7522 0000372A 6650              <1>       push   ax
7523 0000372C 6652              <1>       push   dx
7524 0000372E 66BADA03          <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
7525 00003732 EC                <1>       in     al, dx
7526 00003733 66BAC003          <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
7527 00003737 88D8              <1>       mov    al, bl
7528 00003739 EE                <1>       out    dx, al
7529 0000373A 88F8              <1>       mov    al, bh
7530 0000373C EE                <1>       out    dx, al
7531 0000373D B020              <1>       mov    al, 20h
7532 0000373F EE                <1>       out    dx, al
7533                            <1>       ; ifdef VBOX
7534 00003740 66BADA03          <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
7535 00003744 EC                <1>       in     al, dx
7536                            <1>       ; endif ; VBOX
7537 00003745 665A              <1>       pop    dx
7538 00003747 6658              <1>       pop    ax
7539                            <1> ;no_actl_reg1:
7540 00003749 E909E4FFFF        <1>       jmp    VIDEO_RETURN
7541                            <1>
7542                            <1> set_overscan_border_color:
7543                            <1>       ; 10/08/2016
7544                            <1>       ; Set Overscan/Border Color Register
7545                            <1>       ; BH = 6-bit RGB color to display
7546                            <1>       ;      for that attribute
7547                            <1>
7548 0000374E B311              <1>       mov    bl, 11h
7549 00003750 EBCF              <1>       jmp    short set_single_palette_reg
7550                            <1>
7551                            <1> set_all_palette_reg:
7552                            <1>       ; 10/08/2016
7553                            <1>       ; Set All Palette Registers and Overscan
7554                            <1>       ; EDX = Address of 17 bytes;
7555                            <1>       ; an rgbRGB value for each of 16 palette
7556                            <1>       ; registers plus one for the border.
7557                            <1>
7558 00003752 89D6              <1>       mov    esi, edx ; user buffer
7559 00003754 B911000000        <1>       mov    ecx, 17
7560 00003759 89E7              <1>       mov    edi, esp
7561 0000375B 83EC14            <1>       sub    esp, 20
7562 0000375E E8ABDB0000        <1>       call   transfer_from_user_buffer
7563                            <1>       ;jc    VIDEO_RETURN
7564                            <1>
7565 00003763 66BADA03          <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
7566 00003767 EC                <1>       in     al, dx
7567 00003768 B100              <1>       mov    cl, 0
7568 0000376A 66BAC003          <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
7569                            <1> set_palette_loop:
7570 0000376E 88C8              <1>       mov    al, cl
7571 00003770 EE                <1>       out    dx, al
7572 00003771 8A07              <1>       mov    al, [edi]
7573 00003773 EE                <1>       out    dx, al
7574 00003774 47                <1>       inc    edi
7575 00003775 FEC1              <1>       inc    cl
7576 00003777 80F910            <1>       cmp    cl, 10h
7577 0000377A 75F2              <1>       jne    short set_palette_loop
7578 0000377C B011              <1>       mov    al, 11h
7579 0000377E EE                <1>       out    dx, al
7580 0000377F 8A07              <1>       mov    al, [edi]
7581 00003781 EE                <1>       out    dx, al
7582 00003782 B020              <1>       mov    al, 20h
7583 00003784 EE                <1>       out    dx, al
7584                            <1>       ; ifdef VBOX
7585 00003785 66BADA03          <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
7586 00003789 EC                <1>       in     al, dx
7587                            <1>       ; endif ; VBOX
7588 0000378A 83C414            <1>       add    esp, 20
7589 0000378D E9C5E3FFFF        <1>       jmp    VIDEO_RETURN
7590                            <1>
7591                            <1> toggle_intensity:
7592                            <1>       ; 10/08/2016
7593                            <1>       ; Select Foreground Blink or Bold Background
7594                            <1>       ; BL = 00h = enable bold backgrounds
7595                            <1>       ;          (16 background colors)
7596                            <1>         ;      01h = enable blinking foreground
7597                            <1>       ;          (8 background colors)
7598                            <1>
7599 00003792 66BADA03          <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
7600 00003796 EC                <1>       in     al, dx
7601 00003797 66BAC003          <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
7602 0000379B B010              <1>       mov    al, 10h
7603 0000379D EE                <1>       out    dx, al
7604 0000379E 66BAC103          <1>       mov    dx, 3C1h ; VGAREG_ACTL_READ_DATA
7605 000037A2 EC                <1>       in     al, dx
7606 000037A3 24F7              <1>       and    al, 0F7h
7607 000037A5 80E301            <1>       and    bl, 01h
7608 000037A8 C0E303            <1>       shl    bl, 3
7609 000037AB 08D8              <1>       or     al, bl
7610 000037AD 66BAC003          <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
7611 000037B1 EE                <1>       out    dx, al
7612 000037B2 B020              <1>       mov    al, 20h
7613 000037B4 EE                <1>       out    dx, al
7614                            <1>       ; ifdef VBOX
7615 000037B5 66BADA03          <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
7616 000037B9 EC                <1>       in     al, dx
7617                            <1>       ; endif ; VBOX
7618 000037BA E998E3FFFF        <1>       jmp    VIDEO_RETURN
```

```
                                <1>
                                <1> get_single_palette_reg:
                                <1>       ; 10/08/2016
                                <1>       ; Read One Palette Register
                                <1>         ; INPUT:
                                <1>       ; BL = Palette register to read (00h-0Fh)
                                <1>       ; OUTPUT:
                                <1>       ; BH = Current rgbRGB value of specified register
                                <1>       ;      for that attribute
                                <1>
000037BF 80FB14                 <1>       cmp    bl, 14h
                                <1>       ;ja    short no_actl_reg2
000037C2 0F878FE3FFFF           <1>       ja     VIDEO_RETURN
                                <1>
000037C8 66BADA03               <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
000037CC EC                     <1>       in     al, dx
000037CD 66BAC003               <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
000037D1 88D8                   <1>       mov    al, bl
000037D3 EE                     <1>       out    dx, al
000037D4 66BAC103               <1>       mov    dx, 3C1h ; VGAREG_ACTL_READ_DATA
000037D8 EC                     <1>       in     al, dx
000037D9 8844240D               <1>       mov    [esp+13], al ; bh
000037DD 66BADA03               <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
000037E1 EC                     <1>       in     al, dx
000037E2 66BAC003               <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
000037E6 B020                   <1>       mov    al, 20h
000037E8 EE                     <1>       out    dx, al
                                <1>       ; ifdef VBOX
000037E9 66BADA03               <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
000037ED EC                     <1>       in     al, dx
                                <1>       ; endif ; VBOX
000037EE E964E3FFFF             <1>       jmp    VIDEO_RETURN
                                <1>
                                <1> read_overscan_border_color:
                                <1>       ; 10/08/2016
                                <1>       ; Read Overscan Register
                                <1>       ; OUTPUT:
                                <1>       ; BH = current rgbRGB value
                                <1>       ;      of the overscan/border register
                                <1>
000037F3 B311                   <1>       mov    bl, 11h
000037F5 EBC8                   <1>       jmp    short get_single_palette_reg
                                <1>
                                <1> get_all_palette_reg:
                                <1>       ; 10/08/2016
                                <1>       ; Read All Palette Registers
                                <1>       ; EDX = Address of 17-byte buffer
                                <1>       ;      to receive data
                                <1>
000037F7 89D7                   <1>       mov    edi, edx
000037F9 89E3                   <1>       mov    ebx, esp
000037FB 89DE                   <1>       mov    esi, ebx
000037FD 83EC14                 <1>       sub    esp, 20
                                <1>
00003800 B100                   <1>       mov    cl, 0
                                <1> get_palette_loop:
00003802 66BADA03               <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
00003806 EC                     <1>       in     al, dx
00003807 66BAC003               <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
0000380B 88C8                   <1>       mov    al, cl
0000380D EE                     <1>       out    dx, al
0000380E 66BAC103               <1>       mov    dx, 3C1h ; VGAREG_ACTL_READ_DATA
00003812 EC                     <1>       in     al, dx
00003813 8803                   <1>       mov    [ebx], al
00003815 43                     <1>       inc    ebx
00003816 FEC1                   <1>       inc    cl
00003818 80F910                 <1>       cmp    cl, 10h
0000381B 75E5                   <1>       jne    short get_palette_loop
0000381D 66BADA03               <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
00003821 EC                     <1>       in     al, dx
00003822 66BAC003               <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
00003826 B011                   <1>       mov    al, 11h
00003828 EE                     <1>       out    dx, al
00003829 66BAC103               <1>       mov    dx, 3C1h ; VGAREG_ACTL_READ_DATA
0000382D EC                     <1>       in     al, dx
0000382E 8803                   <1>       mov    [ebx], al
00003830 66BADA03               <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
00003834 EC                     <1>       in     al, dx
00003835 66BAC003               <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
00003839 B020                   <1>       mov    al, 20h
0000383B EE                     <1>       out    dx, al
                                <1>       ; ifdef VBOX
0000383C 66BADA03               <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
00003840 EC                     <1>       in     al, dx
                                <1>       ; endif ; VBOX
                                <1>
00003841 B911000000             <1>       mov    ecx, 17 ; transfer (byte) count
                                <1>       ; ESI = source address in system space
                                <1>       ; EDI = user's buffer address
00003846 E879DA0000             <1>       call   transfer_to_user_buffer
                                <1>
0000384B 83C414                 <1>       add    esp, 20
0000384E E904E3FFFF             <1>       jmp    VIDEO_RETURN
                                <1>
                                <1> set_single_dac_reg:
                                <1>       ; 10/08/2016
                                <1>       ; Set One DAC Color Register
                                <1>       ; BX = color register to set (0-255)
                                <1>         ; CH = green value (00h-3Fh)
                                <1>         ; CL = blue value  (00h-3Fh)
                                <1>         ; DH = red value   (00h-3Fh)
                                <1>
00003853 6652                   <1>       push   dx
00003855 66BAC803               <1>       mov    dx, 3C8h ; VGAREG_DAC_WRITE_ADDRESS
00003859 88D8                   <1>       mov    al, bl
```

```
7724 0000385B EE                <1>       out    dx, al
7725                            <1>       ;mov   dx, 3C9h ; VGAREG_DAC_DATA
7726 0000385C 6642              <1>       inc    dx
7727 0000385E 6658              <1>       pop    ax
7728 00003860 88E0              <1>       mov    al, ah
7729 00003862 EE                <1>       out    dx, al
7730 00003863 88E8              <1>       mov    al, ch
7731 00003865 EE                <1>       out    dx, al
7732 00003866 88C8              <1>       mov    al, cl
7733 00003868 EE                <1>       out    dx, al
7734 00003869 E9E9E2FFFF        <1>       jmp    VIDEO_RETURN
7735                            <1>
7736                            <1> set_all_dac_reg:
7737                            <1>       ; 12/08/2016
7738                            <1>       ; 11/08/2016
7739                            <1>       ; 10/08/2016
7740                            <1>       ; Set a Block of DAC Color Register
7741                            <1>       ; BX = first DAC register to set (0-00FFh)
7742                            <1>       ; ECX = number of registers to set (0-00FFh)
7743                            <1>       ; EDX = addr of a table of R,G,B values
7744                            <1>       ;      (it will be CX*3 bytes long)
7745                            <1>
7746 0000386E 89D6              <1>       mov    esi, edx ; user buffer
7747 00003870 89CA              <1>       mov    edx, ecx
7748 00003872 66D1E1            <1>       shl    cx, 1 ; *2
7749 00003875 01D1              <1>       add    ecx, edx ; ecx = 3*ecx
7750 00003877 89E5              <1>       mov    ebp, esp
7751 00003879 89EF              <1>       mov    edi, ebp
7752 0000387B 29CF              <1>       sub    edi, ecx
7753 0000387D 6683E7FC          <1>       and    di, 0FFFCh ; (dword alignment)
7754 00003881 89FC              <1>       mov    esp, edi
7755 00003883 E886DA0000        <1>       call   transfer_from_user_buffer
7756                            <1>       ;jc    VIDEO_RETURN
7757                            <1>
7758 00003888 89D1              <1>       mov    ecx, edx
7759 0000388A 66BAC803          <1>       mov    dx, 3C8h ; VGAREG_DAC_WRITE_ADDRESS
7760 0000388E 88D8              <1>       mov    al, bl
7761 00003890 EE                <1>       out    dx, al
7762 00003891 66BAC903          <1>       mov    dx, 3C9h ; VGAREG_DAC_DATA
7763                            <1> set_dac_loop:
7764 00003895 8A07              <1>       mov    al, [edi]
7765 00003897 EE                <1>       out    dx, al
7766 00003898 47                <1>       inc    edi
7767 00003899 8A07              <1>       mov    al, [edi]
7768 0000389B EE                <1>       out    dx, al
7769 0000389C 47                <1>       inc    edi
7770 0000389D 8A07              <1>       mov    al, [edi]
7771 0000389F EE                <1>       out    dx, al
7772 000038A0 47                <1>       inc    edi
7773 000038A1 6649              <1>       dec    cx
7774 000038A3 75F0              <1>       jnz    short set_dac_loop
7775 000038A5 89EC              <1>       mov    esp, ebp
7776 000038A7 E9ABE2FFFF        <1>       jmp    VIDEO_RETURN
7777                            <1>
7778                            <1> select_video_dac_color_page:
7779                            <1>       ; 10/08/2016
7780                            <1>       ; DAC Color Paging Functions
7781                            <1>       ; BL = 00H = select color paging mode
7782                            <1>         ;       BH = paging mode
7783                            <1>         ;            00h = 4 blocks of 64 registers
7784                            <1>         ;            01h = 16 blocks of 16 registers
7785                            <1>       ; BL = 01H = activate color page
7786                            <1>         ;       BH = DAC color page number
7787                            <1>         ;            00h-03h (4-page/64-reg mode)
7788                            <1>         ;            00h-0Fh (16-page/16-reg mode)
7789                            <1>
7790 000038AC 66BADA03          <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
7791 000038B0 EC                <1>       in     al, dx
7792 000038B1 66BAC003          <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
7793 000038B5 B010              <1>       mov    al, 10h
7794 000038B7 EE                <1>       out    dx, al
7795 000038B8 66BAC103          <1>       mov    dx, 3C1h ; VGAREG_ACTL_READ_DATA
7796 000038BC EC                <1>       in     al, dx
7797 000038BD 80E301            <1>       and    bl, 01h
7798 000038C0 750E              <1>       jnz    short set_dac_page
7799 000038C2 247F              <1>       and    al, 07Fh
7800 000038C4 C0E707            <1>       shl    bh, 7
7801 000038C7 08F8              <1>       or     al, bh
7802 000038C9 66BAC003          <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
7803 000038CD EE                <1>       out    dx, al
7804 000038CE EB1D              <1>       jmp    short set_actl_normal
7805                            <1> set_dac_page:
7806 000038D0 6650              <1>       push   ax
7807 000038D2 66BADA03          <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
7808 000038D6 EC                <1>       in     al, dx
7809 000038D7 66BAC003          <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
7810 000038DB B014              <1>       mov    al, 14h
7811 000038DD EE                <1>       out    dx, al
7812 000038DE 6658              <1>       pop    ax
7813 000038E0 2480              <1>       and    al, 80h
7814 000038E2 7503              <1>       jnz    short set_dac_16_page
7815 000038E4 C0E702            <1>       shl    bh, 2
7816                            <1> set_dac_16_page:
7817 000038E7 80E70F            <1>       and    bh, 0Fh
7818 000038EA 88F8              <1>       mov    al, bh
7819 000038EC EE                <1>       out    dx, al
7820                            <1> set_actl_normal:
7821 000038ED B020              <1>       mov    al, 20h
7822 000038EF EE                <1>       out    dx, al
7823                            <1>       ; ifdef VBOX
7824 000038F0 66BADA03          <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
7825 000038F4 EC                <1>       in     al, dx
7826                            <1>       ; endif ; VBOX
7827 000038F5 E95DE2FFFF        <1>       jmp    VIDEO_RETURN
7828                            <1>
```

```
                            <1> read_single_dac_reg:
                            <1>       ; 10/08/2016
                            <1>       ; Read One DAC Color Register
                            <1>       ; INPUT:
                            <1>       ; BX = color register to read (0-255)
                            <1>       ; OUTPUT:
                            <1>       ; CH = green value (00h-3Fh)
                            <1>         ; CL = blue value  (00h-3Fh)
                            <1>         ; DH = red value   (00h-3Fh)
                            <1>
000038FA 66BAC703           <1>       mov    dx, 3C7h ; VGAREG_DAC_READ_ADDRESS
000038FE 88D8               <1>       mov    al, bl
00003900 EE                 <1>       out    dx, al
00003901 66BAC903           <1>       mov    dx, 3C9h ; VGAREG_DAC_DATA
00003905 EC                 <1>       in     al, dx
00003906 88442415           <1>       mov    [esp+21], al ; dh
0000390A EC                 <1>       in     al, dx
0000390B 88C5               <1>       mov    ch, al
0000390D EC                 <1>       in     al, dx
0000390E 88C1               <1>       mov    cl, al
00003910 66894C2410         <1>       mov    [esp+16], cx ; cx
00003915 E93DE2FFFF         <1>       jmp    VIDEO_RETURN
                            <1>
                            <1> read_all_dac_reg:
                            <1>       ; 12/08/2016
                            <1>       ; 11/08/2016
                            <1>       ; 10/08/2016
                            <1>       ; Read a Block of DAC Color Registers
                            <1>         ; BX = first DAC register to read (0-00FFh)
                            <1>         ; ECX = number of registers to read (0-00FFh)
                            <1>         ; EDX = addr of a buffer to hold R,G,B values
                            <1>       ;      (CX*3 bytes long)
                            <1>
0000391A 89D7               <1>       mov    edi, edx ; user buffer
0000391C 89CA               <1>       mov    edx, ecx
0000391E 66D1E2             <1>       shl    dx, 1 ; *2
00003921 01CA               <1>       add    edx, ecx ; edx = 3*ecx
00003923 89E5               <1>       mov    ebp, esp
00003925 89EE               <1>       mov    esi, ebp
00003927 29D6               <1>       sub    esi, edx
00003929 6683E6FC           <1>       and    si, 0FFFCh ; (dword alignment)
0000392D 89F4               <1>       mov    esp, esi
0000392F 52                 <1>       push   edx ; 3*ecx
00003930 66BAC703           <1>       mov    dx, 3C7h ; VGAREG_DAC_READ_ADDRESS
00003934 88D8               <1>       mov    al, bl
00003936 EE                 <1>       out    dx, al
00003937 66BAC903           <1>       mov    dx, 3C9h ; VGAREG_DAC_DATA
0000393B 89F3               <1>       mov    ebx, esi
                            <1> read_dac_loop:
0000393D EC                 <1>       in     al, dx
0000393E 8803               <1>       mov    [ebx], al
00003940 43                 <1>       inc    ebx
00003941 EC                 <1>       in     al, dx
00003942 8803               <1>       mov    [ebx], al
00003944 43                 <1>       inc    ebx
00003945 EC                 <1>       in     al, dx
00003946 8803               <1>       mov    [ebx], al
00003948 43                 <1>       inc    ebx
00003949 6649               <1>       dec    cx
0000394B 75F0               <1>       jnz    short read_dac_loop
0000394D 59                 <1>       pop    ecx ; 3*ecx
                            <1>       ; ECX = transfer (byte) count
                            <1>       ; ESI = source address in system space
                            <1>       ; EDI = user's buffer address
0000394E E871D90000         <1>       call   transfer_to_user_buffer
00003953 89EC               <1>       mov    esp, ebp
00003955 E9FDE1FFFF         <1>       jmp    VIDEO_RETURN
                            <1>
                            <1> set_pel_mask:
                            <1>       ; 10/08/2016
                            <1>       ; BL = mask value
0000395A 66BAC603           <1>       mov    dx, 3C6h ; VGAREG_PEL_MASK
0000395E 88D8               <1>       mov    al, bl
00003960 EE                 <1>       out    dx, al
00003961 E9F1E1FFFF         <1>       jmp    VIDEO_RETURN
                            <1>
                            <1> read_pel_mask:
                            <1>       ; 10/08/2016
                            <1>       ; Output: BL = mask value
00003966 66BAC603           <1>       mov    dx, 3C6h ; VGAREG_PEL_MASK
0000396A EC                 <1>       in     al, dx
0000396B 8844240C           <1>       mov    [esp+12], al ; bl
0000396F E9E3E1FFFF         <1>       jmp    VIDEO_RETURN
                            <1>
                            <1> read_video_dac_state:
                            <1>       ; 10/08/2016
                            <1>       ; Query DAC Color Paging State
                            <1>       ; Output:
                            <1>       ; BH = current active DAC color page
                            <1>         ; BL = current active DAC paging mode
                            <1>
00003974 66BADA03           <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
00003978 EC                 <1>       in     al, dx
00003979 66BAC003           <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
0000397D B010               <1>       mov    al, 10h
0000397F EE                 <1>       out    dx, al
00003980 66BAC103           <1>       mov    dx, 3C1h ; VGAREG_ACTL_READ_DATA
00003984 EC                 <1>       in     al, dx
00003985 88C3               <1>       mov    bl, al
00003987 C0EB07             <1>       shr    bl, 7
0000398A 66BADA03           <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
0000398E EC                 <1>       in     al, dx
0000398F 66BAC003           <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
00003993 B014               <1>       mov    al, 14h
00003995 EE                 <1>       out    dx, al
```

```
7934 00003996 66BAC103            <1>       mov    dx, 3C1h ; VGAREG_ACTL_READ_DATA
7935 0000399A EC                  <1>       in     al, dx
7936 0000399B 88C7                <1>       mov    bh, al
7937 0000399D 80E70F              <1>       and    bh, 0Fh
7938 000039A0 F6C301              <1>       test   bl, 01
7939 000039A3 7503                <1>       jnz    short get_dac_16_page
7940 000039A5 C0EF02              <1>       shr    bh, 2
7941                              <1> get_dac_16_page:
7942 000039A8 66BADA03            <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
7943 000039AC EC                  <1>       in     al, dx
7944 000039AD 66BAC003            <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
7945 000039B1 B020                <1>       mov    al, 20h
7946 000039B3 EE                  <1>       out    dx, al
7947                              <1>       ; ifdef VBOX
7948 000039B4 66BADA03            <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
7949 000039B8 EC                  <1>       in     al, dx
7950                              <1>       ; endif ; VBOX
7951 000039B9 66895C240C          <1>       mov    [esp+12], bx ; bx
7952 000039BE E994E1FFFF          <1>       jmp    VIDEO_RETURN
7953                              <1>
7954                              <1> ; 23/11/2020 - TRDOS 386 v2.0.3
7955                              <1> ; VBE 2 BOCHS/QEMU emulator extensions
7956                              <1> ;     for TRDOS 386 v2 kernel (video bios)
7957                              <1>
7958                              <1> ; BOCH/QEMU VBE2 VGA BIOS code
7959                              <1> ;     by Jeroen Janssen (2002)
7960                              <1> ;     by Volker Rupper (2003-2020)
7961                              <1> ; vbe.c (02/01/2020)
7962                              <1>
7963                              <1> ; vbe.h (02/01/2020)
7964                              <1>
7965                              <1> VBE_DISPI_BANK_ADDRESS    equ    0A0000h
7966                              <1> VBE_DISPI_BANK_SIZE_KB    equ    64
7967                              <1>
7968                              <1> VBE_DISPI_MAX_XRES equ    2560
7969                              <1> VBE_DISPI_MAX_YRES equ    1600
7970                              <1>
7971                              <1> VBE_DISPI_IOPORT_INDEX    equ    01CEh
7972                              <1> VBE_DISPI_IOPORT_DATA     equ    01CFh
7973                              <1>
7974                              <1> VBE_DISPI_INDEX_ID equ    00h
7975                              <1> VBE_DISPI_INDEX_XRES      equ    01h
7976                              <1> VBE_DISPI_INDEX_YRES      equ    02h
7977                              <1> VBE_DISPI_INDEX_BPP       equ    03h
7978                              <1> VBE_DISPI_INDEX_ENABLE    equ    04h
7979                              <1> VBE_DISPI_INDEX_BANK      equ    05h
7980                              <1> VBE_DISPI_INDEX_VIRT_WIDTH equ  06h
7981                              <1> VBE_DISPI_INDEX_VIRT_HEIGHT equ 07h
7982                              <1> VBE_DISPI_INDEX_X_OFFSET equ    08h
7983                              <1> VBE_DISPI_INDEX_Y_OFFSET equ    09h
7984                              <1> VBE_DISPI_INDEX_VIDEO_MEMORY_64K equ 0Ah
7985                              <1> VBE_DISPI_INDEX_DDC       equ    0Bh
7986                              <1>
7987                              <1> VBE_DISPI_ID0             equ    0B0C0h
7988                              <1> VBE_DISPI_ID1             equ    0B0C1h
7989                              <1> VBE_DISPI_ID2             equ    0B0C2h
7990                              <1> VBE_DISPI_ID3             equ    0B0C3h
7991                              <1> VBE_DISPI_ID4             equ    0B0C4h
7992                              <1> VBE_DISPI_ID5             equ    0B0C5h
7993                              <1>
7994                              <1> VBE_DISPI_DISABLED equ    00h
7995                              <1> VBE_DISPI_ENABLED  equ    01h
7996                              <1> VBE_DISPI_GETCAPS  equ    02h
7997                              <1> VBE_DISPI_8BIT_DAC equ    20h
7998                              <1> VBE_DISPI_LFB_ENABLED     equ    40h
7999                              <1> VBE_DISPI_NOCLEARMEM      equ    80h
8000                              <1>
8001                              <1> VBE_DISPI_LFB_PHYSICAL_ADDRESS equ 0E0000000h
8002                              <1>
8003                              <1> ; ***
8004                              <1>
8005                              <1> ;// VBE Return Status Info
8006                              <1> ;// AL
8007                              <1> VBE_RETURN_STATUS_SUPPORTED     equ    4Fh
8008                              <1> VBE_RETURN_STATUS_UNSUPPORTED   equ    00h
8009                              <1> ;// AH
8010                              <1> VBE_RETURN_STATUS_SUCCESSFULL   equ    00h
8011                              <1> VBE_RETURN_STATUS_FAILED  equ    01h
8012                              <1> VBE_RETURN_STATUS_NOT_SUPPORTED equ    02h
8013                              <1> VBE_RETURN_STATUS_INVALID equ    03h
8014                              <1>
8015                              <1> ;// VBE Mode Numbers
8016                              <1>
8017                              <1> VBE_MODE_VESA_DEFINED           equ    0100h
8018                              <1> VBE_MODE_REFRESH_RATE_USE_CRTC  equ    0800h
8019                              <1> VBE_MODE_LINEAR_FRAME_BUFFER    equ    4000h
8020                              <1> VBE_MODE_PRESERVE_DISPLAY_MEMORY equ   8000h
8021                              <1>
8022                              <1> ;// Mode Attributes
8023                              <1>
8024                              <1> VBE_MODE_ATTRIBUTE_SUPPORTED              equ 0001h
8025                              <1> VBE_MODE_ATTRIBUTE_EXTENDED_INFO_AVAILABLE equ      0002h
8026                              <1> VBE_MODE_ATTRIBUTE_COLOR_MODE             equ 0008h
8027                              <1> VBE_MODE_ATTRIBUTE_GRAPHICS_MODE   equ 0010h
8028                              <1> VBE_MODE_ATTRIBUTE_LINEAR_FRAME_BUFFER_MODE equ     0080h
8029                              <1> VBE_MODE_ATTRIBUTE_DOUBLE_SCAN_MODE       equ 0100h
8030                              <1> VBE_MODE_ATTRIBUTE_INTERLACE_MODE         equ 0200h
8031                              <1>
8032                              <1> ;// Window attributes
8033                              <1>
8034                              <1> VBE_WINDOW_ATTRIBUTE_RELOCATABLE equ   01h
8035                              <1> VBE_WINDOW_ATTRIBUTE_READABLE    equ   02h
8036                              <1> VBE_WINDOW_ATTRIBUTE_WRITEABLE   equ   04h
8037                              <1>
8038                              <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>
000039C3 55                     <1>       push   ebp ; ***
000039C4 56                     <1>       push   esi ; ****
                                <1>
000039C5 31FF                   <1>       xor    edi, edi  ; mov edi, 0
                                <1>
000039C7 803D[54090000]03       <1>       cmp    byte [vbe3], 3
000039CE 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>
000039D0 66B8014F               <1>       mov    ax, 4F01h
                                <1>
000039D4 E82ADFFFFF             <1>       call   _vbe3_pmfn_return_mode_info
                                <1>
000039D9 6683F84F               <1>       cmp    ax, 004Fh
000039DD 7533                   <1>       jne    short _vbe_rmi_2 ; fail
                                <1>
                                <1>       ; 15/12/2020
                                <1>       ; cx = vbe video mode
000039DF 80E501                 <1>       and    ch, 01h ; clear LFB flag
000039E2 BEFE7B0900             <1>       mov    esi, VBE3MODEINFOBLOCK - 2
000039E7 66890E                 <1>       mov    [esi], cx ; MODEINFO.mode
000039EA E8A6000000             <1>       call   set_lfbinfo_table
000039EF EB22                   <1>       jmp    short _vbe_rmi_3  ; cf = 0
                                <1> _vbe_rmi_1:
000039F1 803D[54090000]02       <1>       cmp    byte [vbe3], 2
000039F8 7219                   <1>       jb     short _vbe_rmi_3 ; cf = 1
000039FA A0[55090000]           <1>       mov    al, [vbe2bios] ; 0C0h-0C5h for emu (*)
000039FF 3CC0                   <1>       cmp    al, 0C0h ; BOCHS/QEMU/VIRTUALBOX (*) ?
00003A01 7210                   <1>       jb     short _vbe_rmi_3  ; cf = 1
00003A03 3CC5                   <1>       cmp    al, 0C5h ; (*)
00003A05 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>
00003A07 E80A000000             <1>       call   vbe_biosfn_return_mode_info
00003A0C 6683F84F               <1>       cmp    ax, 004Fh ; successful ?
00003A10 7401                   <1>       je     short _vbe_rmi_3  ; cf = 0
                                <1> _vbe_rmi_2:
00003A12 F9                     <1>       stc
                                <1>       ; cf = 1
                                <1> _vbe_rmi_3:
00003A13 5E                     <1>       pop    esi ; ****
00003A14 5D                     <1>       pop    ebp ; ***
                                <1>
                                <1>       ;pop   es  ; **
                                <1>       ;pop   ss  ; *
                                <1>
00003A15 C3                     <1>       retn
                                <1>
```

```
8144                                    <1>
8145                                    <1> ; * (TRDOS 386, INT 31h, VESA Video Bios functions)
8146                                    <1> ; * ------------------------------------------------------------
8147                                    <1> ; * Function 01h - Return VBE Mode Information
8148                                    <1> ; * ------------------------------------------------------------
8149                                    <1> ; * Input:
8150                                    <1> ; *         AX = 4F01h
8151                                    <1> ; *         CX = Mode number
8152                                    <1> ; *    (ES:DI) EDI = Pointer to ModeInfoBlock structure
8153                                    <1> ; * Output:
8154                                    <1> ; *         AX = VBE Return Status
8155                                    <1> ; *
8156                                    <1> ; *------------------------------------------------------------
8157                                    <1> ; *
8158                                    <1>
8159                                    <1> vbe_biosfn_return_mode_info:
8160                                    <1>       ; 15/12/2020
8161                                    <1>       ; 14/12/2020
8162                                    <1>       ; 12/12/2020
8163                                    <1>       ; 11/12/2020 (TRDOS 386 v2.0.3)
8164                                    <1>       ;
8165                                    <1>       ; Input:
8166                                    <1>       ;      cx = video (bios) mode
8167                                    <1>       ;      edi = ModeInfoBlock buffer address
8168                                    <1>       ;           (in user's memory space)
8169                                    <1>       ;      (ax = 4F01h)
8170                                    <1>       ; Output:
8171                                    <1>       ;      ax = 004Fh (successful)
8172                                    <1>       ;      ah > 0 -> error
8173                                    <1>       ;
8174                                    <1>       ; Modified registers: esi
8175                                    <1>
8176                                    <1>       ;;push ds  ; *
8177                                    <1>       ;;push es  ; **
8178                                    <1>       ;;push ebp ; ***
8179                                    <1>       ;;push esi ; ****
8180                                    <1>
8181 00003A16 F6C501                   <1>       test   ch, 1
8182 00003A19 7505                     <1>       jnz    short vbe_rmi_1
8183                                    <1>
8184                                    <1>       ; mode number < 100h
8185                                    <1>       ; CGA/VGA mode is not proper this VBE function
8186                                    <1>
8187 00003A1B 29C0                     <1>       sub    eax, eax
8188                                    <1> vbe_rmi_0:
8189                                    <1>       ;mov   ax, 0100h  ; Function is not supported
8190 00003A1D B401                     <1>       mov    ah, 1
8191 00003A1F C3                       <1>       retn
8192                                    <1> vbe_rmi_1:
8193 00003A20 52                       <1>       push   edx ; *****
8194 00003A21 51                       <1>       push   ecx ; ******
8195 00003A22 53                       <1>       push   ebx ; *******
8196 00003A23 57                       <1>       push   edi ; ********
8197                                    <1>
8198                                    <1>       ; 14/12/2020
8199 00003A24 89CB                     <1>       mov    ebx, ecx
8200                                    <1>
8201                                    <1>       ;xor   eax, eax
8202 00003A26 80E7C1                   <1>       and    bh, 0C1h ; use bit 15, 14, 8 only (for bh)
8203 00003A29 883D[1D120300]           <1>       mov    [vbe_mode_x], bh
8204                                    <1>       ;and   bx, 1FFh
8205 00003A2F 80E701                   <1>       and    bh, 1
8206                                    <1>       ;mov   bh, 1
8207                                    <1>
8208                                    <1>       ; Alternative 2 (instead of 'Mode_info_find_mode')
8209 00003A32 E88A060000               <1>       call   set_mode_info_list ; (alternative 2)
8210                                    <1>
8211                                    <1>       ; eax = 0
8212                                    <1>
8213                                    <1>       ;mov   bx, [esi] ; mode
8214                                    <1>
8215                                    <1>       ; Alternative 1 (instead of 'set mode info list')
8216                                    <1>       ;call  mode_info_find_mode ; (alternative 1)
8217                                    <1>
8218 00003A37 09F6                     <1>       or     esi, esi
8219                                    <1>       ; 14/12/2020
8220 00003A39 744D                     <1>       jz     short vbe_rmi_4     ; VBE mode number is wrong
8221                                    <1>                                  ; or it is not supported
8222                                    <1>
8223                                    <1>       ; 15/12/2020
8224                                    <1>       ;mov   bx, [esi] ; mode
8225                                    <1>
8226                                    <1>       ; 12/12/2020
8227                                    <1>       ;call  set_lfbinfo_table
8228                                    <1>
8229 00003A3B F605[1D120300]40         <1>       test   byte [vbe_mode_x], 40h ; LFB model ?
8230 00003A42 7404                     <1>       jz     short vbe_rmi_2
8231                                    <1>
8232 00003A44 C6461C01                 <1>       mov    byte [esi+MODEINFO.NumberOfBanks], 1
8233                                    <1> vbe_rmi_2:
8234                                    <1>       ; (vbe.c, 02/01/2020, vruppert)
8235                                    <1>       ; 11/12/2020 (Erdogan Tan, video.s)
8236                                    <1>       ; Bochs Graphics Adapter
8237                                    <1>       ; vendor_id: 1111h, device id: 1234h
8238                                    <1>
8239 00003A48 E855070000               <1>       call   pci_get_lfb_addr
8240                                    <1>       ;or    eax, eax
8241 00003A4D 7404                     <1>       jz     short vbe_rmi_3
8242                                    <1>       ; zf = 0, ax > 0 (high word of LFB address)
8243                                    <1>       ; set/change LFB address in MODEINFO structure
8244 00003A4F 6689462C                 <1>       mov    [esi+MODEINFO.PhysBasePtr+2], ax
8245                                    <1>       ; 12/12/2020
8246                                    <1>       ;mov   [edi+LFBINFO.LFB_addr+2], ax
8247                                    <1> vbe_rmi_3:
8248                                    <1>       ;test  byte [esi+MODEINFO.WinAAttributes], 1
```

```
8249                                    <1>       ;            ; VBE_WINDOW_ATTRIBUTE_RELOCATABLE = 1
8250                                    <1>         ;jz  short vbe_rmi_4
8251                                    <1>       ;; 11/12/2020
8252                                    <1>       ;; In fact, this is far call address in (Bochs/BGA) Video Bios
8253                                    <1>       ;; Direct user access to kernel subroutines is not possible
8254                                    <1>       ;; in TRDOS 386. Also, TRDOS 386 kernel will support only LFB.
8255                                    <1>       ;; Bank select may be a seperate sysvideo function in future
8256                                    <1>       ;; (if it will be required).
8257                                    <1>       ;mov   dword [esi+MODEINFO.WinFuncPtr], dispi_set_bank_farcall
8258                                    <1> ;vbe_rmi_4:
8259                                    <1>       ; 12/12/2020
8260 00003A53 E83D000000               <1>       call   set_lfbinfo_table
8261                                    <1>
8262                                    <1>       ; 11/12/2020
8263                                    <1>       ; copy 68 bytes of MODE_INFO_LIST to user
8264                                    <1>
8265 00003A58 8B3C24                   <1>       mov    edi, [esp]  ; user's buffer address
8266                                    <1>       ; 12/12/2020
8267 00003A5B 09FF                     <1>       or     edi, edi ; 0 = kernel call
8268                                    <1>                     ; (call from '_vbe_biosfn_return_mode_info')
8269 00003A5D 7432                     <1>       jz     short vbe_rmi_6
8270                                    <1>
8271                                    <1>       ; 15/12/2020
8272                                    <1>       ; prepare 256 bytes MODEINFO buffer at VBE3MODEINFOBLOCK
8273                                    <1>       ; and then, copy buffer conttent to user's buffer
8274 00003A5F 57                       <1>       push   edi
8275 00003A60 BE[3C120300]             <1>       mov    esi, MODE_INFO_LIST + 2    ; MODEINFO.ModeAttributes
8276 00003A65 BF007C0900               <1>       mov    edi, VBE3MODEINFOBLOCK
8277 00003A6A B910000000               <1>       mov    ecx, 66/4  ; 66 bytes
8278 00003A6F F3A5                     <1>       rep    movsd
8279 00003A71 31C0                     <1>       xor    eax, eax
8280 00003A73 B12F                     <1>       mov    cl, (256-68)/4 ; 188 bytes
8281 00003A75 F3AB                     <1>       rep    stosd
8282 00003A77 66AB                     <1>       stosw  ; 2 bytes
8283 00003A79 5F                       <1>       pop    edi
8284 00003A7A BE007C0900               <1>       mov    esi, VBE3MODEINFOBLOCK
8285                                    <1>       ;mov   cx, 256
8286 00003A7F FEC5                     <1>       inc    ch ; cx = 256
8287 00003A81 E83ED80000               <1>       call   transfer_to_user_buffer
8288 00003A86 7309                     <1>       jnc    short vbe_rmi_5
8289                                    <1> vbe_rmi_4:
8290                                    <1>       ;mov   eax, 014Fh ; fail/error
8291 00003A88 31C0                     <1>       xor    eax, eax
8292 00003A8A B401                     <1>       mov    ah, 01h
8293                                    <1>       ;jmp   short vbe_rmi_6
8294 00003A8C E981000000               <1>       jmp    vbe_sm_ret1 ; 11/12/2020
8295                                    <1> vbe_rmi_5:
8296                                    <1>       ; 256 bytes of MODEINFO have been transferred to user
8297                                    <1>       ;mov   eax, 4Fh ; succesfull
8298                                    <1> vbe_rmi_6: ; 12/12/2020
8299 00003A91 31C0                     <1>       xor    eax, eax
8300                                    <1> ;vbe_rmi_6:
8301 00003A93 EB7D                     <1>       jmp    vbe_sm_ret1 ; 11/12/2020
8302                                    <1>
8303                                    <1>       ;pop   edi ; ********
8304                                    <1>       ;pop   ebx ; *******
8305                                    <1>       ;pop   ecx ; ******
8306                                    <1>       ;pop   edx ; *****
8307                                    <1>
8308                                    <1>       ;;pop  esi ; ****
8309                                    <1>       ;;pop  ebp ; ***
8310                                    <1>       ;;pop  es  ; **
8311                                    <1>       ;;pop  ds  ; *
8312                                    <1>
8313                                    <1>       ;retn
8314                                    <1>
8315                                    <1> set_lfbinfo_table:
8316                                    <1>       ; 19/12/2020
8317                                    <1>       ; 11/12/2020
8318                                    <1>       ; Set/Fill LFBINFO structure/table
8319                                    <1>       ;
8320                                    <1>       ; Input:
8321                                    <1>       ;      esi = Mode info list address
8322                                    <1>       ; Output:
8323                                    <1>       ;      LFB_Info address is filled with LFBINFO
8324                                    <1>       ;      edi = LFB_Info address
8325                                    <1>       ;
8326                                    <1>       ; Modified registers: eax, edx (=0), edi
8327                                    <1>
8328 00003A95 BF[2A120300]             <1>       mov    edi, LFB_Info
8329 00003A9A 8B462A                   <1>       mov    eax, [esi+MODEINFO.PhysBasePtr]
8330 00003A9D 894702                   <1>       mov    [edi+LFBINFO.LFB_addr], eax ; LFB address
8331                                    <1>       ;mov   ax, [esi+MODEINFO.mode]
8332 00003AA0 668B06                   <1>       mov    ax, [esi]
8333 00003AA3 668907                   <1>       mov    [edi+LFBINFO.mode],ax
8334 00003AA6 8A461B                   <1>       mov    al, [esi+MODEINFO.BitsPerPixel]
8335 00003AA9 88470E                   <1>       mov    [edi+LFBINFO.bpp], al
8336 00003AAC 29C0                     <1>       sub    eax, eax
8337 00003AAE 668B4614                 <1>       mov    ax, [esi+MODEINFO.XResolution]
8338 00003AB2 6689470A                 <1>       mov    [edi+LFBINFO.X_res], ax
8339 00003AB6 89C2                     <1>       mov    edx, eax ; 19/12/2020
8340 00003AB8 668B4616                 <1>       mov    ax, [esi+MODEINFO.YResolution]
8341 00003ABC 6689470C                 <1>       mov    [edi+LFBINFO.Y_res], ax
8342                                    <1>       ; eax = Y_res ; screen height
8343                                    <1>       ; 19/12/2020
8344 00003AC0 F7E2                     <1>       mul    edx ; X_res*Y_res
8345                                    <1>       ; edx = 0
8346 00003AC2 8A570E                   <1>       mov    dl, [edi+LFBINFO.bpp]
8347                                    <1>       ; Note:
8348                                    <1>       ; Bits per pixel may be 8,16,24,32 for TRDOS 386 v2.
8349                                    <1>       ; (4 bits for pixel is not used for VESA modes here)
8350 00003AC5 C0EA03                   <1>       shr    dl, 3 ; convert bits to byte
8351 00003AC8 F7E2                     <1>       mul    edx
8352                                    <1>       ; eax = screen/page/buffer size in bytes
8353 00003ACA 894706                   <1>       mov    [edi+LFBINFO.LFB_size], eax
```

```
                                <1>       ; edx = 0
                                <1>       ; clear reserved byte in LFBINFO structure/table
00003ACD 88570F                 <1>       mov    [edi+LFBINFO.reserved], dl ; not necessary
00003AD0 C3                     <1>       retn
                                <1>
                                <1> ; * (TRDOS 386, INT 31h, VESA Video Bios functions)
                                <1> ; * ------------------------------------------------------------
                                <1> ; * Function 02h - Set VBE Mode
                                <1> ; * ------------------------------------------------------------
                                <1> ; * Input:
                                <1> ; *         AX = 4F02h
                                <1> ; *         BX = Desired Mode to set
                                <1> ; * Output:
                                <1> ; *         AX = VBE Return Status
                                <1> ; *
                                <1> ; *------------------------------------------------------------
                                <1> ; *
                                <1>
                                <1> vbe_biosfn_set_mode:
                                <1>       ; 12/12/2020
                                <1>       ; 11/12/2020 (LFBINFO table for VESA VBE modes)
                                <1>       ; 27/11/2020
                                <1>       ; 25/11/2020
                                <1>       ; 23/11/2020 (TRDOS 386 v2.0.3)
                                <1>       ; (ref: vbe.c, 02/01/2020, vruppert)
                                <1>       ;
                                <1>       ; Input:
                                <1>       ;     bx = video (bios) mode
                                <1>       ;     ax = 4F02h
                                <1>       ; Output:
                                <1>       ;     ax = 004Fh (successful)
                                <1>       ;     ah > 0 -> error
                                <1>       ;
                                <1>       ; Modified registers: esi
                                <1>
                                <1>       ; 27/11/2020
                                <1>
                                <1>       ;;push ds  ; *
                                <1>       ;;push es  ; **
                                <1>       ;;push ebp ; ***
                                <1>       ;;push esi ; ****
                                <1>
                                <1>       ; 11/12/2020
00003AD1 52                     <1>       push   edx ; *****
00003AD2 51                     <1>       push   ecx ; ******
00003AD3 53                     <1>       push   ebx ; *******
00003AD4 57                     <1>       push   edi ; ********
                                <1>
                                <1>       ;xor   eax, eax
00003AD5 80E7C1                 <1>       and    bh, 0C1h ; use bit 15, 14, 8 only (for bh)
00003AD8 883D[1D120300]         <1>       mov    [vbe_mode_x], bh
00003ADE 80E701                 <1>       and    bh, 1
00003AE1 753C                   <1>       jnz    short vbe_sm_3  ; VESA VBE mode
                                <1>
                                <1>       ;;test bx, 4000h ; VBE_MODE_LINEAR_FRAME_BUFFER
                                <1>       ;test  bh, 40h
                                <1>       ;jz    short vbe_sm_0
                                <1>       ;; lfb_flag
                                <1>       ;mov   al, 40h      ; VBE_DISPI_LFB_ENABLED
                                <1> vbe_sm_0:
                                <1>       ; 27/11/2020
00003AE3 B080                   <1>       mov    al, 80h
                                <1>       ;test  bh, 80h ; VBE_MODE_PRESERVE_DISPLAY_MEMORY
                                <1>       ;jnz   short vbe_sm_1 ; no_clear
                                <1>       ;; clear
                                <1>       ;sub   al, al ; 0
00003AE5 8405[1D120300]         <1>       test   [vbe_mode_x], al ; 80h
00003AEB 7402                   <1>       jz     short vbe_sm_1 ; clear display memory
                                <1>       ; no_clear
00003AED 08C3                   <1>       or     bl, al ; VBE_MODE_PRESERVE_DISPLAY_MEMORY
                                <1> vbe_sm_1:
                                <1>       ; check non vesa mode
                                <1>       ;;cmp  bx, 100h ; VBE_MODE_VESA_DEFINED
                                <1>       ;;jna  short vbe_sm_2
                                <1>       ;and   bh, 1
                                <1>       ;jnz   short vbe_sm_3
                                <1>
                                <1>       ; BX <= 1FFh
                                <1>
                                <1>       ; 27/11/2020
                                <1>       ;or    bl, al ; al = 80h if no_clear option is set
                                <1>       ;             ; al = 0 if no_clear option is not set
                                <1>
                                <1>       ; 25/11/2020
                                <1>       ; VBE DISPI will be disabled in 'biosfn_set_video_mode'
                                <1>
                                <1>       ;xor   al, al ; 0 ; VBE_DISPI_DISABLED
                                <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
00003AEF 31C0           <1>       xor    eax, eax ; 0
00003AF1 E8A3040000     <1>       call   dispi_set_enable
                        <1>
00003AF6 88D8           <1>       mov    al, bl
                        <1>       ;jmp   _set_mode ; (in 'biosfn_set_video_mode' sub)
00003AF8 E86BE0FFFF     <1>       call   _set_mode ; will return with cf=1 only if
                        <1>                        ; desired mode is not implemented
                        <1>       ; _retn: return from subroutine
00003AFD 721A           <1>       jc     short vbe_sm_2 ; 25/11/2020
                        <1>
                        <1>       ; 26/11/2020
00003AFF 31C0           <1>       xor    eax, eax
00003B01 A0[BA6F0000]   <1>       mov    al, [CRT_MODE]
                        <1>       ; 27/11/2020
00003B06 8A25[278E0100] <1>       mov    ah, [noclearmem] ; 80h or 0
                        <1>       ;and   ah 80h
00003B0C 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
00003B12 B04F           <1>       mov    al, 4Fh ; Function call successful
                        <1>       ; eax = 004Fh
                        <1> vbe_sm_ret2:
                        <1>       ; 11/12/2020
00003B14 5F             <1>       pop    edi ; ********
00003B15 5B             <1>       pop    ebx ; *******
00003B16 59             <1>       pop    ecx ; ******
00003B17 5A             <1>       pop    edx ; *****
                        <1>
                        <1>       ;;pop  esi ; ****
                        <1>       ;;pop  ebp ; ***
                        <1>       ;;pop  es  ; **
                        <1>       ;;pop  ds  ; *
                        <1>
00003B18 C3             <1>       retn
                        <1>
                        <1> vbe_sm_2:
                        <1>       ;mov   ax, 0100h ; Function is not supported
                        <1>       ; 27/11/2020
00003B19 31C0           <1>       xor    eax, eax
00003B1B B401           <1>       mov    ah, 01h
                        <1>       ; eax = 0100h
                        <1>       ;retn
00003B1D 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
00003B1F 803D[BA6F0000]03 <1>     cmp    byte [CRT_MODE], 03h
00003B26 75BB           <1>       jne    short vbe_sm_0
00003B28 E8F0E2FFFF     <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
00003B2D 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>
00003B32 09F6           <1>       or     esi, esi
00003B34 74E3           <1>       jz     short vbe_sm_2 ; VBE mode number is wrong
                        <1>                             ; or it is not supported
                        <1>
                        <1>       ; 11/12/2020
00003B36 668B1E         <1>       mov    bx, [esi] ; mode
                        <1>
                        <1>       ; 27/11/2020
00003B39 0A3D[1D120300] <1>       or     bh, [vbe_mode_x]
                        <1>
                        <1>       ; save VESA VBE mode
00003B3F 66891D[1E120300] <1>     mov    [video_mode], bx
                        <1>              ; 27/11/2020
                        <1>              ; bit 0 to 8 = VESA VBE mode
                        <1>              ; bit 9 to 13 = 0 (bit 0 to 13 = mode)
                        <1>              ; bit 14 = Linear/Flat Frame Buffer flag
                        <1>              ; bit 15 = 'memory not cleared
                        <1>              ;         at last mode set' flag
                        <1>
                        <1>       ; first disable current mode
                        <1>       ; (when switching between vesa modes)
                        <1>       ; 'dispi_set_enable(VBE_DISPI_DISABLED);'
                        <1>
                        <1>       ;mov   ax, VBE_DISPI_DISABLED ; 0
00003B46 29C0           <1>       sub    eax, eax ; 0
                        <1>
00003B48 E84C040000     <1>       call   dispi_set_enable
```

```
                                <1>
                                <1>       ; 11/12/2020
00003B4D 8A461B                 <1>       mov   al, [esi+MODEINFO.BitsPerPixel]
                                <1>       ; ah = 0
                                <1>
                                <1>       ;cmp   byte [esi+MODEINFO.BitsPerPixel], 8
00003B50 3C08                   <1>       cmp   al, 8
00003B52 750B                   <1>       jne   short vbe_sm_5
                                <1>
                                <1>       ; 11/12/2020
                                <1>       ;push  edi
00003B54 50                     <1>       push  eax
                                <1>       ; 'load_dac_palette(3);'
00003B55 56                     <1>       push  esi
00003B56 B403                   <1>       mov   ah, 3  ; palette3, 256 colors
00003B58 E828F1FFFF             <1>       call  load_dac_palette
00003B5D 5E                     <1>       pop   esi
                                <1>       ; 11/12/2020
00003B5E 58                     <1>       pop   eax
                                <1>       ;pop   edi
                                <1> vbe_sm_5:
                                <1>       ;'dispi_set_bpp(cur_info->info.BitsPerPixel);'
                                <1>       ; 11/12/2020 (al = bits per pixel, ah = 0)
                                <1>       ;xor   ah, ah
                                <1>       ;mov   al, [esi+MODEINFO.BitsPerPixel]
00003B5F E849040000             <1>       call  dispi_set_bpp
                                <1>         ;'dispi_set_xres(cur_info->info.XResolution);'
00003B64 668B4614               <1>       mov   ax, [esi+MODEINFO.XResolution]
00003B68 E846040000             <1>       call  dispi_set_xres
                                <1>         ;'dispi_set_yres(cur_info->info.YResolution);'
00003B6D 668B4616               <1>       mov   ax, [esi+MODEINFO.YResolution]
00003B71 E843040000             <1>       call  dispi_set_yres
                                <1>
                                <1>       ;'dispi_set_bank(0);'
                                <1>       ;xor   ax, ax
00003B76 31C0                   <1>       xor   eax, eax ; 0
00003B78 E842040000             <1>       call  dispi_set_bank
                                <1>         ;'dispi_set_enable(VBE_DISPI_ENABLED|no_clear|lfb_flag);'
                                <1>       ;mov   ax, di
                                <1>       ; ah = 0 ; 27/11/2020
00003B7D A0[1D120300]           <1>       mov   al, [vbe_mode_x] ; restore VBE mode bit 14 & 15
00003B82 0C01                   <1>       or    al, 1 ; VBE_DISPI_ENABLED
00003B84 E810040000             <1>       call  dispi_set_enable
                                <1>
                                <1>         ; 'vga_compat_setup();'
00003B89 E846040000             <1>       call  vga_compat_setup
                                <1>
                                <1>       ; 11/12/2020
00003B8E E802FFFFFF             <1>       call  set_lfbinfo_table
                                <1>
                                <1>       ; 26/11/2020
00003B93 31C0                   <1>       xor   eax, eax
00003B95 FEC8                   <1>       dec   al
00003B97 A2[BA6F0000]           <1>       mov   [CRT_MODE], al ; 0FFh = VESA VBE mode sign
                                <1>
                                <1>       ; 27/11/2020
00003B9C E971FFFFFF             <1>       jmp   vbe_sm_ret1 ; Function call successful
                                <1>
                                <1>       ; 27/11/2020
                                <1>       ;mov   al, 4Fh
                                <1>       ;     ; eax = 004Fh = Function call successful
                                <1>       ;jmp   short vbe_sm_ret2
                                <1>
                                <1> ; * (TRDOS 386, INT 31h, VESA Video Bios functions)
                                <1> ; * ------------------------------------------------------
                                <1> ; * Function 03h - Return Current VBE Mode
                                <1> ; * ------------------------------------------------------
                                <1> ; * Input:
                                <1> ; *         AX = 4F03h
                                <1> ; * Output:
                                <1> ; *         AX = VBE Return Status
                                <1> ; *         BX = Current VBE Mode
                                <1> ; *
                                <1> ; *------------------------------------------------------
                                <1> ; *
                                <1>
                                <1> vbe_biosfn_return_current_mode:
                                <1>       ; 11/12/2020
                                <1>       ; 27/11/2020 (TRDOS 386 v2.0.3)
                                <1>       ; (ref: vbe.c, 02/01/2020, vruppert)
                                <1>       ;
                                <1>       ; Input:
                                <1>       ;     none
                                <1>       ; Output:
                                <1>       ;     ax = 004Fh (successful)
                                <1>       ;     ah > 0 -> error
                                <1>       ;     bx = current video (bios) mode (if ah = 0)
                                <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>
```

```
8669 00003BA1 A0[BA6F0000]        <1>       mov    al, [CRT_MODE] ; current cga/vga mode
8670 00003BA6 3CFF                <1>       cmp    al, 0FFh ; VBE extension signature
8671 00003BA8 720E                <1>       jb     short vbe_gm_1 ; get CGA/VGA mode
8672                              <1>
8673                              <1>       ; get VBE mode
8674                              <1> vbe_gm_0:
8675 00003BAA 66A1[1E120300]      <1>       mov    ax, [video_mode]
8676                              <1>              ; BX bits:
8677                              <1>              ; bit 0 to 8 = VESA VBE video mode
8678                              <1>              ; bit 9 to 13 = 0
8679                              <1>              ; bit 14 = last mode set LFB option
8680                              <1>              ;         1 - linear/flat frame buffer
8681                              <1>              ;         0 - windowed frame buffer
8682                              <1>              ; bit 15 = last mode set no_clear option
8683                              <1>              ;         0 - video memory cleared
8684                              <1>              ;         1 - video memory not cleared
8685                              <1>
8686                              <1> vbe_gm_return:
8687                              <1>       ;pop   edx ; ******
8688 00003BB0 0FB7D8              <1>       movzx  ebx, ax
8689                              <1> ;vbe_srs_retn:
8690 00003BB3 31C0                <1>       xor    eax, eax ; 0
8691 00003BB5 B04F                <1>       mov    al, 4Fh ; ax = 004Fh (successful)
8692 00003BB7 C3                  <1>       retn
8693                              <1>
8694                              <1> vbe_gm_1:
8695                              <1>       ; legacy (old, standard) CGA/VGA bios video mode
8696 00003BB8 8A25[278E0100]      <1>       mov    ah, [noclearmem] ; 80h or 0
8697                              <1>              ; BX bits:
8698                              <1>              ; bit 0 to 7 = video mode
8699                              <1>              ; bit 8 to 13 = 0
8700                              <1>              ; bit 14 = 0 (not LFB mode) CGA/VGA
8701                              <1>              ; bit 15 = 1 if [noclearmem] = 80h
8702                              <1>              ;         0 if [noclearmem] = 0
8703 00003BBE EBF0                <1>       jmp    short vbe_gm_return
8704                              <1>
8705                              <1> ; * (TRDOS 386, INT 31h, VESA Video Bios functions)
8706                              <1> ; * ------------------------------------------------------
8707                              <1> ; * Function 04h - Save/Restore State
8708                              <1> ; * ------------------------------------------------------
8709                              <1> ; * Input:
8710                              <1> ; *          AX = 4F04h
8711                              <1> ; *              DL = 00h Return Save/Restore State buff size
8712                              <1> ; *                   01h Save State
8713                              <1> ; *                   02h Restore State
8714                              <1> ; *              CX = Requested states
8715                              <1> ; *                bit 0 - controller hardware state
8716                              <1> ; *                bit 1 - BIOS data state
8717                              <1> ; *                bit 2 - DAC state
8718                              <1> ; *                bit 3 - register state
8719                              <1> ; *     (ES:BX) EBX = Pointer to buffer (if DL <> 00h)
8720                              <1> ; * Output:
8721                              <1> ; *          AX = VBE Return Status
8722                              <1> ; *          BX = Number of 64-byte blocks
8723                              <1> ; *               to hold the state buffer (if DL=00h)
8724                              <1> ; *
8725                              <1> ; *------------------------------------------------------
8726                              <1> ; *
8727                              <1>
8728                              <1> vbe_biosfn_save_restore_state:
8729                              <1>       ; 23/01/2021
8730                              <1>       ; 16/01/2021
8731                              <1>       ; 14/01/2021
8732                              <1>       ; 13/01/2021
8733                              <1>       ; 12/01/2021
8734                              <1>       ; 11/01/2021 (TRDOS 386 v2.0.3)
8735                              <1>       ; (ref: vbe.c, 02/01/2020, vruppert)
8736                              <1>       ;
8737                              <1>       ; Input:
8738                              <1>       ;      dl = sub function
8739                              <1>       ;      cl = requested state
8740                              <1>       ;      ebx = pointer to buffer (if dl<>00h)
8741                              <1>       ; Output:
8742                              <1>       ;      ax = 004Fh (successful)
8743                              <1>       ;      ah > 0 -> error
8744                              <1>       ;      bx = Number of 64-byte blocks
8745                              <1>       ;           to hold the state buffer (if DL=00h)
8746                              <1>
8747                              <1>       ; Modified registers: eax, ebx, edi
8748                              <1>
8749                              <1>       ; 14/01/2021
8750 00003BC0 09DB                <1>       or     ebx, ebx ; user's buffer address
8751 00003BC2 750A                <1>       jnz    short _vbe_biosfn_save_restore_state
8752                              <1>
8753 00003BC4 20D2                <1>       and    dl, dl
8754 00003BC6 7406                <1>       jz     short _vbe_biosfn_save_restore_state
8755                              <1>
8756                              <1>       ; function failed
8757                              <1>       ;mov   eax, 0100h
8758                              <1>       ;xor   eax, eax
8759                              <1>       ;inc   ah  ; eax = 0100h
8760                              <1>       ; 16/01/2021
8761 00003BC8 B84F010000          <1>       mov    eax, 014Fh
8762 00003BCD C3                  <1>       retn
8763                              <1>
8764                              <1> _vbe_biosfn_save_restore_state:
8765                              <1>       ; 23/01/2021
8766                              <1>       ; 14/01/2021
8767                              <1>       ; ebx = 0 if the caller is kernel ('sysvideo')
8768                              <1>
8769                              <1>       ; 13/01/2021
8770 00003BCE 57                  <1>       push   edi
8771 00003BCF 52                  <1>       push   edx
8772 00003BD0 51                  <1>       push   ecx
8773                              <1>
```

```
                                <1>       ; 23/01/2021
                                <1>       ; 12/01/2021
00003BD1 80FA02                 <1>       cmp    dl, 2
00003BD4 7757                   <1>       ja     short vbe_srs_7 ; 23/01/2021
                                <1>                      ; invalid sub function
00003BD6 83F90F                 <1>       cmp    ecx, 0Fh
00003BD9 7752                   <1>       ja     short vbe_srs_7 ; invalid !
                                <1>
00003BDB 20D2                   <1>       and    dl, dl
00003BDD 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]
00003BDF 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:
00003BE4 6683C33F               <1>       add    bx, 63
00003BE8 66C1EB06               <1>       shr    bx, 6 ; / 64
                                <1>
                                <1> vbe_srs_retn:
00003BEC 31C0                   <1>       xor    eax, eax ; 0
                                <1> vbe_srs_0:  ; 16/01/2021
00003BEE B04F                   <1>       mov    al, 4Fh ; ax = 004Fh (successful)
                                <1> ;vbe_srs_0:
                                <1>       ; 13/01/2021
00003BF0 59                     <1>       pop    ecx
00003BF1 5A                     <1>       pop    edx
00003BF2 5F                     <1>       pop    edi
                                <1>
00003BF3 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
00003BF4 80E10F                 <1>       and    cl, 0Fh ; 8, 4, 2, 1
00003BF7 7434                   <1>       jz     short vbe_srs_7 ; cx = 0 -> invalid !
                                <1>
00003BF9 BF00760900             <1>       mov    edi, VBE3SAVERESTOREBLOCK
                                <1>
00003BFE 80FA01                 <1>       cmp    dl, 1
00003C01 7730                   <1>       ja     short vbe_srs_8
                                <1>
                                <1>       ; save video state
                                <1>
00003C03 F6C107                 <1>       test   cl, 07h ; 4, 2, 1
00003C06 740A                   <1>       jz     short vbe_srs_5  ; vbe dispi regs state
                                <1>
00003C08 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
00003C0D F6C108                 <1>       test   cl, 8
00003C10 7405                   <1>       jz     short vbe_srs_6
                                <1> vbe_srs_5:
00003C12 E8AC010000             <1>       call   vbe_biosfn_save_video_state
                                <1>       ; edi = end position
                                <1>       ;      in VBE3SAVERESTOREBLOCK
                                <1>       ;      (VGA save_state offset)
                                <1>       ; modified regs: edi, eax, edx, ch
                                <1> vbe_srs_6:
                                <1>       ; 23/01/2021
00003C17 21DB                   <1>       and    ebx, ebx
00003C19 74D1                   <1>       jz     short vbe_srs_retn ; the caller is kernel
                                <1>
00003C1B BE00760900             <1>       mov    esi, VBE3SAVERESTOREBLOCK
00003C20 29F7                   <1>       sub    edi, esi
00003C22 89F9                   <1>       mov    ecx, edi ; transfer count in bytes
                                <1>
                                <1>       ;; 14/01/2021
```

```
                                <1>       ;and   ebx, ebx
                                <1>       ;jz    short vbe_srs_10 ; the caller is kernel
                                <1>
00003C24 89DF                   <1>       mov    edi, ebx ; user's buffer address
00003C26 E899D60000             <1>       call   transfer_to_user_buffer
00003C2B 73BF                   <1>       jnc    short vbe_srs_retn
                                <1> vbe_srs_7:
                                <1>       ; // function failed
                                <1>       ;mov   eax, 0100h
00003C2D 31C0                   <1>       xor    eax, eax
00003C2F FEC4                   <1>       inc    ah  ; eax = 0100h
                                <1>       ; 16/01/2021
                                <1>       ; ax = 0014Fh
                                <1>       ;retn
                                <1>       ; 13/01/2021
00003C31 EBBB                   <1>       jmp    short vbe_srs_0
                                <1> vbe_srs_8:
                                <1>       ;cmp   dl, 2
                                <1>       ;jne   short vbe_srs_7
                                <1>       ;            ; invalid sub function
                                <1>
                                <1>       ; 14/01/2021
00003C33 09DB                   <1>       or     ebx, ebx ; user's buffer address
                                <1>       ;jnz   short vbe_srs_11
                                <1>
                                <1>       ; the caller is kernel ('sysvideo')
                                <1>       ;jmp   short vbe_srs_12
                                <1>       ; 23/01/2021
00003C35 7414                   <1>       jz     short vbe_srs_12 ; 'sysvideo' call
                                <1> vbe_srs_11:
00003C37 89DE                   <1>       mov    esi, ebx ; user's buffer address
                                <1>       ; 23/01/2021
                                <1>       ;push  ebx
                                <1>
00003C39 E827000000             <1>       call   vbe_srs_gbs
                                <1>
                                <1>       ; restore video state
                                <1>
                                <1>       ;mov   edi, VBE3SAVERESTOREBLOCK
00003C3E 51                     <1>       push   ecx
00003C3F 89D9                   <1>       mov    ecx, ebx ; transfer count in bytes
00003C41 E8C8D60000             <1>       call   transfer_from_user_buffer
00003C46 59                     <1>       pop    ecx
                                <1>       ; 23/01/2021
                                <1>       ;pop   ebx
00003C47 89F3                   <1>       mov    ebx, esi
00003C49 72E2                   <1>       jc     short vbe_srs_7
                                <1>
                                <1> vbe_srs_12:
                                <1>       ;mov   esi, VBE3SAVERESTOREBLOCK
00003C4B 89FE                   <1>       mov    esi, edi
                                <1>
00003C4D F6C107                 <1>       test   cl, 07h ; 4, 2, 1
00003C50 740C                   <1>       jz     short vbe_srs_9     ; vbe dispi regs state
                                <1>
00003C52 E8A8010000             <1>       call   biosfn_restore_video_state
00003C57 72D4                   <1>       jc     short vbe_srs_7 ; invalid buffer content !
                                <1>       ; esi = current position
                                <1>       ;      in VBE3SAVERESTOREBLOCK
                                <1>       ;      (VGA save_state offset)
                                <1>       ; modified regs: esi, eax, edx, ch
00003C59 F6C108                 <1>       test   cl, 8
                                <1>       ;jz    short vbe_srs_10
                                <1>       ; 23/01/2020
00003C5C EB8E                   <1>       jmp    short vbe_srs_retn
                                <1> vbe_srs_9:
00003C5E E8F8020000             <1>       call   vbe_biosfn_restore_video_state
                                <1>
                                <1>       ; modified regs: esi, eax, edx, ch
                                <1>
00003C63 EB87                   <1>       jmp    short vbe_srs_retn
                                <1>
                                <1> ;vbe_srs_10:
                                <1> ;     ; successful
                                <1> ;     xor    eax, eax ; 0
                                <1> ;     mov    al, 4Fh ; ax = 004Fh (successful)
                                <1> ;     retn
                                <1>
                                <1> vbe_srs_gbs:
                                <1>       ; return buffer size according to flags
00003C65 89CB                   <1>       mov    ebx, ecx ; options/flags
00003C67 D0E3                   <1>       shl    bl, 1
00003C69 668B9B[713C0000]       <1>       mov    bx, [ebx+vbestatebufsize]
00003C70 C3                     <1>       retn
                                <1>
                                <1> vbestatebufsize:
                                <1>       ; ---------------------------------------
                                <1>       ; CL = 0  1   2    3    4    5    6    7
                                <1>       ; ---------------------------------------
00003C71 0000460028006E0004-    <1>       dw     0, 70, 40, 110, 772, 842, 812, 882
00003C7A 034A032C037203         <1>
                                <1>       ; ---------------------------------------
                                <1>       ; CL = 8   9   10  11   12   13   14   15
                                <1>       ; ---------------------------------------
00003C81 120058003A00800016-    <1>       dw     18, 88, 58, 128, 790, 860, 830, 900
00003C8A 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
```

```
8982                                    <1> VGAREG_SEQU_ADDRESS       equ 3C4h
8983                                    <1> VGAREG_SEQU_DATA   equ 3C5h
8984                                    <1>
8985                                    <1> VGAREG_PEL_MASK           equ 3C6h
8986                                    <1> VGAREG_DAC_STATE   equ 3C7h
8987                                    <1> VGAREG_DAC_READ_ADDRESS   equ 3C7h
8988                                    <1> VGAREG_DAC_WRITE_ADDRESS equ 3C8h
8989                                    <1> VGAREG_DAC_DATA           equ 3C9h
8990                                    <1>
8991                                    <1> VGAREG_READ_FEATURE_CTL   equ 3CAh
8992                                    <1> VGAREG_READ_MISC_OUTPUT   equ 3CCh
8993                                    <1>
8994                                    <1> VGAREG_GRDC_ADDRESS       equ 3CEh
8995                                    <1> VGAREG_GRDC_DATA   equ 3CFh
8996                                    <1>
8997                                    <1> ;VGAREG_MDA_CRTC_ADDRESS equ 3B4h
8998                                    <1> ;VGAREG_MDA_CRTC_DATA     equ 3B5h
8999                                    <1> VGAREG_VGA_CRTC_ADDRESS   equ 3D4h
9000                                    <1> VGAREG_VGA_CRTC_DATA      equ 3D5h
9001                                    <1>
9002                                    <1> ;VGAREG_MDA_WRITE_FEATURE_CTL equ 3BAh
9003                                    <1> VGAREG_VGA_WRITE_FEATURE_CTL equ 3DAh
9004                                    <1> VGAREG_ACTL_RESET  equ 3DAh
9005                                    <1>
9006                                    <1> ;VGAREG_MDA_MODECTL       equ 3B8h
9007                                    <1> VGAREG_CGA_MODECTL equ 3D8h
9008                                    <1> VGAREG_CGA_PALETTE equ 3D9h
9009                                    <1>
9010                                    <1> biosfn_save_video_state:
9011                                    <1>       ; 22/01/2021
9012                                    <1>       ; 12/01/2021
9013                                    <1>       ; 11/01/2021 (TRDOS 386 v2.0.3)
9014                                    <1>       ; (vgabios.c)
9015                                    <1>
9016                                    <1>       ; modified registers: eax, edx, edi, ch
9017                                    <1>
9018                                    <1>       ;mov  edi, VBE3SAVERESTOREBLOCK
9019                                    <1>
9020                                    <1>       ; input: edi = state buffer address
9021                                    <1>
9022 00003C91 F6C101                   <1>       test  cl, 1
9023 00003C94 0F8485000000             <1>       jz    bfn_svs_4
9024                                    <1>
9025 00003C9A 66BAC403                 <1>       mov   dx, VGAREG_SEQU_ADDRESS ; 3C7h
9026 00003C9E EC                       <1>       in    al, dx
9027 00003C9F AA                       <1>       stosb
9028                                    <1>       ;mov  dx, VGAREG_VGA_CRTC_ADDRESS ; 3D4h
9029 00003CA0 B2D4                     <1>       mov   dl, 0D4h
9030 00003CA2 EC                       <1>       in    al, dx
9031 00003CA3 AA                       <1>       stosb
9032                                    <1>       ;mov  dx, VGAREG_GRDC_ADDRESS ; 3CEh
9033 00003CA4 B2CE                     <1>         mov dl, 0CEh
9034 00003CA6 EC                       <1>       in    al, dx
9035 00003CA7 AA                       <1>       stosb
9036                                    <1>       ;mov  dx, VGAREG_ACTL_RESET ; 3DAh
9037 00003CA8 B2DA                     <1>       mov   dl, 0DAh
9038 00003CAA EC                       <1>       in    al, dx
9039                                    <1>       ;mov  dx, VGAREG_ACTL_ADDRESS ; 3C0h
9040 00003CAB B2C0                     <1>       mov   dl, 0C0h
9041 00003CAD EC                       <1>       in    al, dx
9042 00003CAE AA                       <1>       stosb
9043 00003CAF 88C4                     <1>       mov   ah, al ; ar_index
9044                                    <1>       ;mov  dx, VGAREG_READ_FEATURE_CTL ; 3CAh
9045 00003CB1 B2CA                     <1>       mov   dl, 0CAh
9046 00003CB3 EC                       <1>       in    al, dx
9047 00003CB4 AA                       <1>       stosb
9048                                    <1>       ; (5 bytes are writen above)
9049                                    <1>
9050                                    <1>       ; for(i=1;i<=4;i++){
9051 00003CB5 B001                     <1>       mov   al, 1
9052                                    <1>       ;;mov  dx, VGAREG_SEQU_ADDRESS ; 3C4h
9053                                    <1>       ;mov  dl, 0C4h
9054 00003CB7 B504                     <1>       mov   ch, 4
9055                                    <1> bfn_svs_0:
9056                                    <1>       ; outb(VGAREG_SEQU_ADDRESS, i);
9057                                    <1>       ;mov  dx, VGAREG_SEQU_ADDRESS ; 3C4h
9058 00003CB9 B2C4                     <1>       mov   dl, 0C4h
9059 00003CBB EE                       <1>       out   dx, al
9060                                    <1>       ;mov  dx, VGAREG_SEQU_DATA  ; 3C5h
9061 00003CBC FEC2                     <1>       inc   dl  ; dx = 3C5h
9062                                    <1>       ; inb(VGAREG_SEQU_DATA)
9063 00003CBE 50                       <1>       push  eax
9064 00003CBF EC                       <1>       in    al, dx
9065 00003CC0 AA                       <1>       stosb ; (4 bytes in loop)
9066 00003CC1 58                       <1>       pop   eax
9067                                    <1>       ;mov  dx, VGAREG_SEQU_ADDRESS ; 3C4h
9068                                    <1>       ;dec  dl
9069 00003CC2 FEC0                     <1>       inc   al  ; i++
9070 00003CC4 FECD                     <1>       dec   ch
9071 00003CC6 75F1                     <1>       jnz   short bfn_svs_0
9072                                    <1>
9073                                    <1>       ; outb(VGAREG_SEQU_ADDRESS, 0);
9074 00003CC8 28C0                     <1>       sub   al, al ; 0
9075 00003CCA EE                       <1>       out   dx, al
9076                                    <1>       ; inb(VGAREG_SEQU_DATA)
9077                                    <1>       ;mov  dx, VGAREG_SEQU_DATA ; 3C5h
9078 00003CCB FEC2                     <1>       inc   dl  ; dx = 3C5h
9079 00003CCD EC                       <1>       in    al, dx
9080 00003CCE AA                       <1>       stosb ; (+1 byte)
9081                                    <1>
9082                                    <1>         ; for(i=0;i<=0x18;i++) {
9083 00003CCF 28C0                     <1>       sub   al, al ; 0
9084                                    <1>       ;;mov  dx, VGAREG_VGA_CRTC_ADDRESS ; 3D4h
9085                                    <1>       ;mov  dl, 0D4h
9086 00003CD1 B519                     <1>       mov   ch, 25
```

```
9087                            <1> bfn_svs_1:
9088                            <1>         ; outb(crtc_addr,i);
9089                            <1>       ;mov   dx, VGAREG_VGA_CRTC_ADDRESS ; 3D4h
9090 00003CD3 B2D4             <1>       mov    dl, 0D4h
9091 00003CD5 EE               <1>       out    dx, al
9092                            <1>       ;mov   dx, VGAREG_VGA_CRTC_DATA ; 3D5h
9093 00003CD6 FEC2             <1>       inc    dl  ; dx = 3D5h
9094                            <1>       ; inb(crtc_addr+1)
9095 00003CD8 50               <1>       push   eax
9096 00003CD9 EC               <1>       in     al, dx
9097 00003CDA AA               <1>       stosb  ; (25 bytes in loop)
9098 00003CDB 58               <1>       pop    eax
9099                            <1>       ;mov   dx, VGAREG_VGA_CRTC_ADDRESS ; 3D4h
9100                            <1>       ;dec   dl
9101 00003CDC FEC0             <1>       inc    al  ; i++
9102 00003CDE FECD             <1>       dec    ch
9103 00003CE0 75F1             <1>       jnz    short bfn_svs_1
9104                            <1>
9105 00003CE2 80E420           <1>       and    ah, 20h  ; (ar_index & 0x20)
9106                            <1>         ; for(i=0;i<=0x13;i++) {
9107 00003CE5 28C0             <1>       sub    al, al ; 0
9108 00003CE7 B514             <1>       mov    ch, 20
9109                            <1> bfn_svs_2:
9110                            <1>       ; inb(VGAREG_ACTL_RESET);
9111                            <1>       ;mov   dx, VGAREG_ACTL_RESET ; 3DAh
9112 00003CE9 B2DA             <1>       mov    dl, 0DAh
9113 00003CEB 50               <1>       push   eax
9114 00003CEC EC               <1>       in     al, dx
9115 00003CED 8A0424           <1>       mov    al, [esp]
9116                            <1>       ; outb(VGAREG_ACTL_ADDRESS, i | (ar_index & 0x20));
9117 00003CF0 08E0             <1>       or     al, ah
9118                            <1>       ;mov   dx, VGAREG_ACTL_ADDRESS ; 3C0h
9119 00003CF2 B2C0             <1>       mov    dl, 0C0h
9120 00003CF4 EE               <1>       out    dx, al
9121                            <1>       ;mov   dx, VGAREG_ACTL_READ_DATA ; 3C1h
9122                            <1>       ;mov   dl, 0C1h
9123 00003CF5 FEC2             <1>       inc    dl
9124 00003CF7 EC               <1>       in     al, dx
9125 00003CF8 AA               <1>       stosb  ; (20 bytes in loop)
9126 00003CF9 58               <1>       pop    eax
9127 00003CFA FEC0             <1>       inc    al  ; i++
9128 00003CFC FECD             <1>       dec    ch
9129 00003CFE 75E9             <1>       jnz    short bfn_svs_2
9130                            <1>
9131                            <1>       ; inb(VGAREG_ACTL_RESET);
9132                            <1>       ;mov   dx, VGAREG_ACTL_RESET ; 3DAh
9133 00003D00 B2DA             <1>       mov    dl, 0DAh
9134 00003D02 EC               <1>       in     al, dx
9135                            <1>
9136                            <1>         ; for(i=0;i<=8;i++) {
9137 00003D03 28C0             <1>       sub    al, al ; 0
9138                            <1>       ;;mov  dx, VGAREG_GRDC_ADDRESS ; 3CEh
9139                            <1>       ;mov   dl, 0CEh
9140 00003D05 B509             <1>       mov    ch, 9
9141                            <1> bfn_svs_3:
9142                            <1>       ; outb(VGAREG_GRDC_ADDRESS,i)
9143                            <1>       ;mov   dx, VGAREG_GRDC_ADDRESS ; 3CEh
9144 00003D07 B2CE             <1>       mov    dl, 0CEh
9145 00003D09 EE               <1>       out    dx, al
9146                            <1>       ; inb(VGAREG_ACTL_READ_DATA)
9147 00003D0A 50               <1>       push   eax
9148                            <1>       ;mov   dx, VGAREG_GRDC_DATA ; 3CFh
9149                            <1>       ;mov   dl, 0CFh
9150 00003D0B FEC2             <1>       inc    dl
9151 00003D0D EC               <1>       in     al, dx
9152 00003D0E AA               <1>       stosb  ; (9 bytes in loop)
9153 00003D0F 58               <1>       pop    eax
9154                            <1>       ;dec   dl
9155 00003D10 FEC0             <1>       inc    al  ; i++
9156 00003D12 FECD             <1>       dec    ch
9157 00003D14 75F1             <1>       jnz    short bfn_svs_3
9158                            <1>
9159                            <1>       ; write_word(ES, BX, crtc_addr); BX+= 2;
9160                            <1>       ; (offset 64)
9161 00003D16 66B8D403         <1>       mov    ax, 3D4h ; VGAREG_VGA_CRTC_ADDRESS
9162 00003D1A 66AB             <1>       stosw  ; (2 bytes (1 word))
9163                            <1>
9164                            <1>         ; /* XXX: read plane latches */
9165 00003D1C 31C0             <1>       xor    eax, eax  ; 0
9166 00003D1E AB               <1>              stosd  ; (4 bytes)
9167                            <1>
9168                            <1>       ; (total 70 bytes are written above as controller hardware state)
9169                            <1>
9170                            <1> bfn_svs_4:
9171                            <1>       ; 12/01/2021 (TRDOS 386 v2.0.3)
9172 00003D1F F6C102           <1>       test   cl, 2
9173 00003D22 7476             <1>       jz     short bfn_svs_6
9174                            <1>
9175                            <1>       ; VIDEO BIOS DATA
9176                            <1>       ; !!! this data is valid for TRDOS 386 v2 kernel only !!!
9177                            <1>       ; (this is not same with BOCHS/PLEX86 video bios, BIOS data)
9178                            <1>
9179                            <1>       ; if (CX & 2) {
9180                            <1>         ;write_byte(ES, BX, read_byte(BIOSMEM_SEG,BIOSMEM_CURRENT_MODE)); BX++;
9181                            <1>         ;write_word(ES, BX, read_word(BIOSMEM_SEG,BIOSMEM_NB_COLS)); BX += 2;
9182                            <1>         ;write_word(ES, BX, read_word(BIOSMEM_SEG,BIOSMEM_PAGE_SIZE)); BX += 2;
9183                            <1>         ;write_word(ES, BX, read_word(BIOSMEM_SEG,BIOSMEM_CRTC_ADDRESS)); BX += 2;
9184                            <1>         ;write_byte(ES, BX, read_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS)); BX++;
9185                            <1>         ;write_word(ES, BX, read_word(BIOSMEM_SEG,BIOSMEM_CHAR_HEIGHT)); BX += 2;
9186                            <1>         ;write_byte(ES, BX, read_byte(BIOSMEM_SEG,BIOSMEM_VIDEO_CTL)); BX++;
9187                            <1>         ;write_byte(ES, BX, read_byte(BIOSMEM_SEG,BIOSMEM_SWITCHES)); BX++;
9188                            <1>         ;write_byte(ES, BX, read_byte(BIOSMEM_SEG,BIOSMEM_MODESET_CTL)); BX++;
9189                            <1>         ;write_word(ES, BX, read_word(BIOSMEM_SEG,BIOSMEM_CURSOR_TYPE)); BX += 2;
9190                            <1>         ;for(i=0;i<8;i++) {
9191                            <1>         ;   write_word(ES, BX, read_word(BIOSMEM_SEG, BIOSMEM_CURSOR_POS+2*i));
```

```
9192                                    <1>         ;   BX += 2;
9193                                    <1>         ;}
9194                                    <1>         ;write_word(ES, BX, read_word(BIOSMEM_SEG,BIOSMEM_CURRENT_START)); BX += 2;
9195                                    <1>         ;write_byte(ES, BX, read_byte(BIOSMEM_SEG,BIOSMEM_CURRENT_PAGE)); BX++;
9196                                    <1>         ;/* current font */
9197                                    <1>         ;write_word(ES, BX, read_word(0, 0x1f * 4)); BX += 2;
9198                                    <1>         ;write_word(ES, BX, read_word(0, 0x1f * 4 + 2)); BX += 2;
9199                                    <1>         ;write_word(ES, BX, read_word(0, 0x43 * 4)); BX += 2;
9200                                    <1>         ;write_word(ES, BX, read_word(0, 0x43 * 4 + 2)); BX += 2;
9201                                    <1>
9202                                    <1>       ; !!! save TRDOS 386 v2 kernel spesific video bios data !!!
9203                                    <1>       ; (which is/are used by 'SET_MODE' function and/or it's sub functions)
9204                                    <1>
9205 00003D24 66B8D403                 <1>       mov   ax, 3D4h ; CRTC_ADDR, always 3D4h (color VGA) for TRDOS 386 v2
9206 00003D28 66AB                     <1>       stosw
9207 00003D2A A0[BA6F0000]             <1>       mov   al, [CRT_MODE] ; Current video mode (0FFh for VESA VBE modes)
9208 00003D2F AA                       <1>       stosb
9209 00003D30 A0[BB6F0000]             <1>       mov   al, [CRT_MODE_SET] ; 29h for mode 03h ; TRDOS 386 feature only !
9210 00003D35 AA                       <1>       stosb
9211 00003D36 66A1[1E120300]           <1>       mov   ax, [video_mode] ; Current VESA VBE (SVGA, extended VGA) mode
9212 00003D3C 66AB                     <1>       stosw               ; (valid if [CRT_MODE] = 0FFh)
9213 00003D3E 66A1[288E0100]           <1>       mov   ax, [CRT_LEN] ; page size (in bytes)
9214 00003D44 66AB                     <1>       stosw
9215 00003D46 66A1[AC810100]           <1>       mov   ax, [CRT_START] ; video page start offset
9216 00003D4C 66AB                     <1>       stosw
9217 00003D4E A0[BC6F0000]             <1>       mov   al, [CRT_COLS] ; nbcols, characters per row
9218 00003D53 AA                       <1>       stosb
9219 00003D54 A0[C26F0000]             <1>       mov   al, [VGA_ROWS] ; nbrows, (character) rows per page (not rows-1)
9220 00003D59 AA                       <1>       stosb
9221 00003D5A A0[BE6F0000]             <1>       mov   al, [CHAR_HEIGHT] ; character font height (8 or 16 or 14)
9222 00003D5F AA                       <1>       stosb
9223 00003D60 A0[BF6F0000]             <1>       mov   al, [VGA_VIDEO_CTL] ; ROM BIOS DATA AREA Offset 87h
9224 00003D65 AA                       <1>       stosb
9225 00003D66 A0[C06F0000]             <1>       mov   al, [VGA_SWITCHES] ; feature bit switches
9226 00003D6B AA                       <1>       stosb
9227 00003D6C A0[C16F0000]             <1>       mov   al, [VGA_MODESET_CTL] ; basic mode set options
9228 00003D71 AA                       <1>       stosb
9229                                    <1>       ; followings are only used by TRDOS 386 v2 (IBM PC/AT ROMBIOS) code
9230                                    <1>       ; (bochs/plex86 does not use and return those)
9231 00003D72 A0[BD6F0000]             <1>       mov   al, [CRT_PALETTE] ; current color palette ; TRDOS 386 feature only !
9232 00003D77 AA                       <1>       stosb
9233 00003D78 A0[BE810100]             <1>       mov   al, [ACTIVE_PAGE] ; current video page
9234 00003D7D AA                       <1>       stosb
9235 00003D7E 66A1[D36F0000]           <1>       mov   ax, [CURSOR_MODE] ; cursor type
9236 00003D84 66AB                     <1>       stosw
9237                                    <1>       ;mov  eax, [CURSOR_POSN] ; cursor position for video page 0 and 1
9238                                    <1>       ;stosd
9239                                    <1>       ;mov  eax, [CURSOR_POSN+4] ; cursor position for video page 2 and 3
9240                                    <1>       ;stosd
9241                                    <1>       ;mov  eax, [CURSOR_POSN+8] ; cursor position for video page 4 and 5
9242                                    <1>       ;stosd
9243                                    <1>       ;mov  eax, [CURSOR_POSN+12] ; cursor position for video page 6 and 7
9244                                    <1>       ;stosd
9245 00003D86 56                       <1>       push  esi
9246 00003D87 B504                     <1>       mov   ch, 4
9247 00003D89 BE[AE810100]             <1>       mov   esi, CURSOR_POSN
9248                                    <1> bfn_svs_5:
9249 00003D8E A5                       <1>       movsd
9250 00003D8F FECD                     <1>       dec   ch
9251 00003D91 75FB                     <1>       jnz   short bfn_svs_5
9252 00003D93 5E                       <1>       pop   esi
9253                                    <1>       ; (font addr) protected mode address in kernel's/system memory space
9254                                    <1>       ; (not accessable/meaningful address value by user)
9255 00003D94 A1[3A8E0100]             <1>       mov   eax, [VGA_INT43H] ; VGA current (default) font address
9256 00003D99 AB                       <1>       stosd
9257                                    <1>
9258                                    <1>       ; (total 40 bytes are written above as BIOS data state)
9259                                    <1>
9260                                    <1> bfn_svs_6:
9261                                    <1>       ; 12/01/2021
9262 00003D9A F6C104                   <1>       test  cl, 4
9263 00003D9D 7423                     <1>       jz    short bfn_svs_8
9264                                    <1>
9265                                    <1>       ;/* XXX: check this */
9266                                    <1>             ; /* read/write mode dac */
9267                                    <1>         ;write_byte(ES, BX, inb(VGAREG_DAC_STATE)); BX++;
9268                                    <1>       ;     ; /* pix address */
9269                                    <1>         ;write_byte(ES, BX, inb(VGAREG_DAC_WRITE_ADDRESS)); BX++;
9270                                    <1>         ;write_byte(ES, BX, inb(VGAREG_PEL_MASK)); BX++;
9271                                    <1>         ;// Set the whole dac always, from 0
9272                                    <1>         ;outb(VGAREG_DAC_WRITE_ADDRESS,0x00);
9273                                    <1>         ;for(i=0;i<256*3;i++) {
9274                                    <1>         ;   write_byte(ES, BX, inb(VGAREG_DAC_DATA)); BX++;
9275                                    <1>         ;}
9276                                    <1>         ;write_byte(ES, BX, 0); BX++; /* color select register */
9277                                    <1>
9278                                    <1>       ; /* read/write mode dac */
9279 00003D9F 66BAC703                 <1>       mov   dx, 3C7h ; VGAREG_DAC_STATE
9280 00003DA3 EC                       <1>       in    al, dx
9281 00003DA4 AA                       <1>       stosb
9282                                    <1>       ; /* pix address */
9283                                    <1>       ;mov  dx, VGAREG_DAC_WRITE_ADDRESS ; 3C8h
9284                                    <1>       ;mov  dl, 0C8h
9285 00003DA5 FEC2                     <1>       inc   dl
9286 00003DA7 EC                       <1>       in    al, dx
9287 00003DA8 AA                       <1>       stosb
9288                                    <1>       ;mov  dx, VGAREG_PEL_MASK  ; 3C6h
9289 00003DA9 B2C6                     <1>       mov   dl, 0C6h
9290 00003DAB EC                       <1>       in    al, dx
9291 00003DAC AA                       <1>       stosb
9292                                    <1>       ;// Set the whole dac always, from 0
9293 00003DAD 30C0                     <1>       xor   al, al ; 0
9294                                    <1>       ;mov  dx, VGAREG_DAC_WRITE_ADDRESS ; 3C8h
9295 00003DAF B2C8                     <1>       mov   dl, 0C8h
9296 00003DB1 EE                       <1>       out   dx, al
```

```
                                <1>
00003DB2 51                     <1>       push   ecx ; 22/01/2021
                                <1>       ;for(i=0;i<256*3;i++) {
00003DB3 B900030000             <1>       mov    ecx, 256*3 ; 768 bytes
                                <1>       ;mov   dx, VGAREG_DAC_DATA ; 3C9h
                                <1>       ;mov   dl, 0C9h
00003DB8 FEC2                   <1>       inc    dl ; dx = 3C9h
                                <1> bfn_svs_7:
00003DBA EC                     <1>       in     al, dx
00003DBB AA                     <1>       stosb
00003DBC E2FC                   <1>       loop   bfn_svs_7
00003DBE 59                     <1>       pop    ecx ; 22/01/2021
                                <1>
                                <1>       ; /* color select register */
00003DBF 28C0                   <1>       sub    al, al ; 0
00003DC1 AA                     <1>       stosb
                                <1>
                                <1>       ; (total 772 bytes are written above as DAC state)
                                <1> bfn_svs_8:
00003DC2 C3                     <1>       retn
                                <1>
                                <1> vbe_biosfn_save_video_state:
                                <1>       ; 23/01/2021
                                <1>       ; 13/01/2021
                                <1>       ; 12/01/2021 (TRDOS 386 v2.0.3)
                                <1>       ; (vbe.c)
                                <1>
                                <1>       ; modified registers: eax, edx, edi, ch
                                <1>
                                <1>       ; input: edi = state buffer address
                                <1>       ; output:
                                <1>       ;      VBE DISPI register contents will be saved
                                <1>       ;      (18 bytes, 9 words)
                                <1>
                                <1>       ; outw(VBE_DISPI_IOPORT_INDEX,VBE_DISPI_INDEX_ENABLE);
                                <1>       ; enable = inw(VBE_DISPI_IOPORT_DATA);
                                <1>       ; write_word(ES, BX, enable);
                                <1>       ; BX += 2;
                                <1>       ; if (!(enable & VBE_DISPI_ENABLED))
                                <1>       ;     return;
                                <1>       ; for(i = VBE_DISPI_INDEX_XRES;
                                <1>       ;             i <= VBE_DISPI_INDEX_Y_OFFSET; i++) {
                                <1>       ;    if (i != VBE_DISPI_INDEX_ENABLE) {
                                <1>       ;        outw(VBE_DISPI_IOPORT_INDEX, i);
                                <1>       ;        write_word(ES, BX, inw(VBE_DISPI_IOPORT_DATA));
                                <1>       ;    BX += 2;
                                <1>       ;        }
                                <1>       ; }
                                <1>
00003DC3 66BACE01               <1>       mov    dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
00003DC7 B804000000             <1>       mov    eax, 04h  ; VBE_DISPI_INDEX_ENABLE
00003DCC 66EF                   <1>       out    dx, ax
                                <1>       ;mov   dx, 01CFh ; VBE_DISPI_IOPORT_DATA
00003DCE FEC2                   <1>       inc    dl
00003DD0 66ED                   <1>       in     ax, dx ; enable (status)
00003DD2 66AB                   <1>       stosw
00003DD4 6683E001               <1>       and    ax, 1 ; VBE_DISPI_ENABLED
00003DD8 7505                   <1>       jnz    short vbe_bfn_svs_0
                                <1>       ; 23/01/2021
                                <1>       ; ax = 0
                                <1>       ; VBE_DISPI_DISABLED
                                <1>       ; 13/01/2021
                                <1>       ; clear remain 8 bytes
                                <1>       ;xor   eax, eax
00003DDA AB                     <1>       stosd  ; 2
00003DDB AB                     <1>       stosd  ; 2
00003DDC AB                     <1>       stosd  ; 2
00003DDD AB                     <1>       stosd  ; 2
00003DDE C3                     <1>       retn
                                <1> vbe_bfn_svs_0:
                                <1>       ; VBE_DISPI_ENABLED
                                <1>
                                <1>       ;sub   eax, eax
00003DDF 28C0                   <1>       sub    al, al ; eax = 0
                                <1>
                                <1>       ; from VBE_DISPI_INDEX_XRES
                                <1>       ;   to VBE_DISPI_INDEX_BPP
                                <1>
00003DE1 B503                   <1>       mov    ch, 3
                                <1>       ; al = 0 ; VBE_DISPI_INDEX_XRES - 1
                                <1>
00003DE3 E804000000             <1>       call   vbe_bfn_svs_1
                                <1>
                                <1>       ; from VBE_DISPI_INDEX_BANK
                                <1>       ;   to VBE_DISPI_INDEX_Y_OFFSET
                                <1>
00003DE8 FEC0                   <1>       inc    al
                                <1>       ; al = 4 ; VBE_DISPI_INDEX_BANK - 1
                                <1>
00003DEA B505                   <1>       mov    ch, 5
                                <1> vbe_bfn_svs_1:
00003DEC FEC0                   <1>       inc    al ; from VBE_DISPI_INDEX_XRES
                                <1>                  ;   to VBE_DISPI_INDEX_BPP
                                <1>       ;mov   dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
00003DEE FECA                   <1>       dec    dl ; 1CEh
00003DF0 66EF                   <1>       out    dx, ax
00003DF2 50                     <1>       push   eax
                                <1>       ;mov   dx, 01CFh ; VBE_DISPI_IOPORT_DATA
00003DF3 FEC2                   <1>       inc    dl ; 1CFh
00003DF5 66ED                   <1>       in     ax, dx
00003DF7 66AB                   <1>       stosw
00003DF9 58                     <1>       pop    eax
00003DFA FECD                   <1>       dec    ch
00003DFC 75EE                   <1>       jnz    short vbe_bfn_svs_1
00003DFE C3                     <1>       retn
```

```
9402                                    <1>
9403                                    <1> ; 11/01/2021
9404                                    <1> VGAREG_ACTL_ADDRESS       equ 3C0h
9405                                    <1> VGAREG_ACTL_WRITE_DATA    equ 3C0h
9406                                    <1> VGAREG_ACTL_READ_DATA     equ 3C1h
9407                                    <1>
9408                                    <1> VGAREG_INPUT_STATUS       equ 3C2h
9409                                    <1> VGAREG_WRITE_MISC_OUTPUT equ 3C2h
9410                                    <1> VGAREG_VIDEO_ENABLE       equ 3C3h
9411                                    <1> VGAREG_SEQU_ADDRESS       equ 3C4h
9412                                    <1> VGAREG_SEQU_DATA   equ 3C5h
9413                                    <1>
9414                                    <1> VGAREG_PEL_MASK           equ 3C6h
9415                                    <1> VGAREG_DAC_STATE   equ 3C7h
9416                                    <1> VGAREG_DAC_READ_ADDRESS   equ 3C7h
9417                                    <1> VGAREG_DAC_WRITE_ADDRESS equ 3C8h
9418                                    <1> VGAREG_DAC_DATA           equ 3C9h
9419                                    <1>
9420                                    <1> VGAREG_READ_FEATURE_CTL   equ 3CAh
9421                                    <1> VGAREG_READ_MISC_OUTPUT   equ 3CCh
9422                                    <1>
9423                                    <1> VGAREG_GRDC_ADDRESS       equ 3CEh
9424                                    <1> VGAREG_GRDC_DATA   equ 3CFh
9425                                    <1>
9426                                    <1> ;VGAREG_MDA_CRTC_ADDRESS equ 3B4h
9427                                    <1> ;VGAREG_MDA_CRTC_DATA     equ 3B5h
9428                                    <1> VGAREG_VGA_CRTC_ADDRESS   equ 3D4h
9429                                    <1> VGAREG_VGA_CRTC_DATA      equ 3D5h
9430                                    <1>
9431                                    <1> ;VGAREG_MDA_WRITE_FEATURE_CTL equ 3BAh
9432                                    <1> VGAREG_VGA_WRITE_FEATURE_CTL equ 3DAh
9433                                    <1> VGAREG_ACTL_RESET  equ 3DAh
9434                                    <1>
9435                                    <1> ;VGAREG_MDA_MODECTL       equ 3B8h
9436                                    <1> VGAREG_CGA_MODECTL equ 3D8h
9437                                    <1> VGAREG_CGA_PALETTE equ 3D9h
9438                                    <1>
9439                                    <1> biosfn_restore_video_state:
9440                                    <1>       ; 22/01/2021
9441                                    <1>       ; 13/01/2021
9442                                    <1>       ; 12/01/2021 (TRDOS 386 v2.0.3)
9443                                    <1>       ; (vgabios.c)
9444                                    <1>
9445                                    <1>       ; modified registers: eax, edx, esi, edi, ch
9446                                    <1>
9447                                    <1>       ;mov   esi, VBE3SAVERESTOREBLOCK
9448                                    <1>
9449                                    <1>       ; input: esi = state buffer address
9450                                    <1>
9451 00003DFF F6C101                   <1>       test   cl, 1
9452 00003E02 0F84A9000000             <1>       jz     bfn_rvs_6
9453                                    <1>
9454 00003E08 66817E40D403             <1>       cmp    word [esi+64], 3D4h ; must be 3D4h
9455 00003E0E 7402                     <1>       je     short bfn_rvs_0
9456                                    <1>                   ; it is seen as valid buffer
9457 00003E10 F9                       <1>       stc
9458 00003E11 C3                       <1>       retn
9459                                    <1>
9460                                    <1> bfn_rvs_0:
9461 00003E12 89F7                     <1>       mov    edi, esi ; addr1
9462 00003E14 83C605                   <1>       add    esi, 5 ; skip 1st 5 bytes for now
9463                                    <1>
9464                                    <1>       ; // Reset Attribute Ctl flip-flop
9465                                    <1>         ; inb(VGAREG_ACTL_RESET);
9466 00003E17 66BADA03                 <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
9467 00003E1B EC                       <1>       in     al, dx
9468                                    <1>
9469                                    <1>       ; for(i=1;i<=4;i++){
9470 00003E1C B001                     <1>       mov    al, 1
9471                                    <1>       ;;mov  dx, VGAREG_SEQU_ADDRESS ; 3C4h
9472                                    <1>       ;mov   dl, 0C4h
9473 00003E1E B504                     <1>       mov    ch, 4
9474                                    <1> bfn_rvs_1:
9475                                    <1>       ; outb(VGAREG_SEQU_ADDRESS, i);
9476                                    <1>       ;mov   dx, VGAREG_SEQU_ADDRESS ; 3C4h
9477 00003E20 B2C4                     <1>       mov    dl, 0C4h
9478 00003E22 EE                       <1>       out    dx, al
9479                                    <1>       ;mov   dx, VGAREG_SEQU_DATA  ; 3C5h
9480 00003E23 FEC2                     <1>       inc    dl  ; dx = 3C5h
9481                                    <1>       ; outb(VGAREG_SEQU_DATA)
9482 00003E25 50                       <1>       push   eax
9483 00003E26 AC                       <1>       lodsb  ; (4 bytes in loop)
9484 00003E27 EE                       <1>       out    dx, al
9485 00003E28 58                       <1>       pop    eax
9486                                    <1>       ;mov   dx, VGAREG_SEQU_ADDRESS ; 3C4h
9487                                    <1>       ;dec   dl
9488 00003E29 FEC0                     <1>       inc    al  ; i++
9489 00003E2B FECD                     <1>       dec    ch
9490 00003E2D 75F1                     <1>       jnz    short bfn_rvs_1
9491                                    <1>
9492                                    <1>       ; outb(VGAREG_SEQU_ADDRESS, 0);
9493 00003E2F 28C0                     <1>       sub    al, al ; 0
9494 00003E31 EE                       <1>       out    dx, al
9495                                    <1>       ; outb(VGAREG_SEQU_DATA)
9496                                    <1>       ;mov   dx, VGAREG_SEQU_DATA ; 3C5h
9497 00003E32 FEC2                     <1>       inc    dl  ; dx = 3C5h
9498 00003E34 AC                       <1>       lodsb  ; (+1 byte)
9499 00003E35 EE                       <1>       out    dx, al
9500                                    <1>
9501                                    <1>       ; // Disable CRTC write protection
9502                                    <1>       ; outw(crtc_addr,0x0011);
9503                                    <1>       ;mov   dx, VGAREG_VGA_CRTC_ADDRESS ; 3D4h
9504 00003E36 B2D4                     <1>       mov    dl, 0D4h
9505 00003E38 66B81100                 <1>       mov    ax, 11h
9506 00003E3C 66EF                     <1>       out    dx, ax
```

```
9507                            <1>
9508                            <1>       ; // Set CRTC regs
9509                            <1>
9510                            <1>       ; for(i=0;i<=0x18;i++) {
9511                            <1>         ;   if (i != 0x11) {
9512 00003E3E 28C0             <1>       sub   al, al ; 0
9513                            <1>       ;;mov  dx, VGAREG_VGA_CRTC_ADDRESS ; 3D4h
9514                            <1>       ;mov   dl, 0D4h
9515 00003E40 B519             <1>       mov   ch, 25
9516                            <1> bfn_rvs_2:
9517                            <1>         ; outb(crtc_addr,i);
9518                            <1>       ;mov   dx, VGAREG_VGA_CRTC_ADDRESS ; 3D4h
9519 00003E42 B2D4             <1>       mov   dl, 0D4h
9520 00003E44 EE               <1>       out   dx, al
9521                            <1>       ;mov   dx, VGAREG_VGA_CRTC_DATA ; 3D5h
9522 00003E45 FEC2             <1>       inc   dl  ; dx = 3D5h
9523                            <1>       ; inb(crtc_addr+1)
9524 00003E47 50               <1>       push  eax
9525 00003E48 AC               <1>       lodsb ; (25 bytes in loop)
9526 00003E49 EE               <1>       out   dx, al
9527 00003E4A 58               <1>       pop   eax
9528                            <1>       ;mov   dx, VGAREG_VGA_CRTC_ADDRESS ; 3D4h
9529                            <1>       ;dec   dl
9530 00003E4B FEC0             <1>       inc   al  ; i++
9531 00003E4D 3C11             <1>       cmp   al, 17 ; 11h
9532 00003E4F 7505             <1>       jne   short bfn_rvs_3
9533 00003E51 AC               <1>       lodsb
9534 00003E52 88C4             <1>       mov   ah, al ; *
9535 00003E54 B012             <1>       mov   al, 18
9536                            <1> bfn_rvs_3:
9537 00003E56 FECD             <1>       dec   ch
9538 00003E58 75E8             <1>       jnz   short bfn_rvs_2
9539                            <1>
9540                            <1>       ; // select crtc base address
9541                            <1>         ; v = inb(VGAREG_READ_MISC_OUTPUT) & ~0x01;
9542                            <1>       ;if (crtc_addr = 0x3d4)
9543                            <1>       ;   v |= 0x01;
9544                            <1>       ; outb(VGAREG_WRITE_MISC_OUTPUT, v);
9545                            <1>
9546                            <1>       ;;mov  dx, VGAREG_READ_MISC_OUTPUT ; 3CCh
9547                            <1>       ;mov   dl, 0CCh
9548                            <1>       ;in    al, dl
9549                            <1>       ;and   al, 1
9550                            <1>       ;;mov  dx, VGAREG_WRITE_MISC_OUTPUT ; 3C2h
9551                            <1>       ;mov   dl, 0C2h
9552                            <1>       ;or    al, 1
9553                            <1>       ;out   dx, al
9554                            <1>
9555                            <1>       ; // enable write protection if needed
9556                            <1>       ;outb(crtc_addr, 0x11);
9557                            <1>         ;outb(crtc_addr+1, read_byte(ES, BX - 0x18 + 0x11));
9558                            <1>       ;mov   dx, VGAREG_VGA_CRTC_ADDRESS ; 3D4h
9559 00003E5A B2D4             <1>       mov   dl, 0D4h
9560 00003E5C B011             <1>       mov   al, 11h
9561 00003E5E EE               <1>       out   dx, al
9562 00003E5F 88E0             <1>       mov   al, ah ; *
9563 00003E61 FEC2             <1>       inc   dl ; dx = 3D5h
9564 00003E63 EE               <1>       out   dx, al
9565                            <1>
9566                            <1>       ; // Set Attribute Ctl
9567 00003E64 8A6703           <1>       mov   ah, [edi+3] ; addr1+3, ah = ar_index
9568 00003E67 80E420           <1>       and   ah, 20h  ; (ar_index & 0x20)
9569                            <1>
9570                            <1>         ; inb(VGAREG_ACTL_RESET);
9571                            <1>       ;mov   dx, 3DAh ; VGAREG_ACTL_RESET
9572 00003E6A B2DA             <1>       mov   dl, 0DAh
9573 00003E6C EC               <1>       in    al, dx
9574                            <1>
9575                            <1>         ; for(i=0;i<=0x13;i++) {
9576 00003E6D 28C0             <1>       sub   al, al ; 0
9577 00003E6F B514             <1>       mov   ch, 20
9578                            <1> bfn_rvs_4:
9579                            <1>         ; outb(VGAREG_ACTL_ADDRESS, i | (ar_index & 0x20));
9580 00003E71 50               <1>       push  eax
9581 00003E72 08E0             <1>       or    al, ah
9582                            <1>       ;mov   dx, VGAREG_ACTL_ADDRESS ; 3C0h
9583 00003E74 B2C0             <1>       mov   dl, 0C0h
9584 00003E76 EE               <1>       out   dx, al
9585                            <1>       ;mov   dx, VGAREG_ACTL_WRITE_DATA ; 3C0h
9586                            <1>       ;mov   dl, 0C0h
9587 00003E77 AC               <1>       lodsb ; (20 bytes in loop)
9588 00003E78 EE               <1>       out   dx, al
9589 00003E79 58               <1>       pop   eax
9590 00003E7A FEC0             <1>       inc   al  ; i++
9591 00003E7C FECD             <1>       dec   ch
9592 00003E7E 75F1             <1>       jnz   short bfn_rvs_4
9593                            <1>
9594                            <1>       ; outb(VGAREG_ACTL_ADDRESS, ar_index);
9595                            <1>       ;mov   dx, VGAREG_ACTL_ADDRESS ; 3C0h
9596                            <1>       ;mov   dl, 0C0h
9597 00003E80 88E0             <1>       mov   al, ah ; ar_index
9598 00003E82 EE               <1>       out   dx, al
9599                            <1>
9600                            <1>       ; inb(VGAREG_ACTL_RESET);
9601                            <1>       ;mov   dx, VGAREG_ACTL_RESET ; 3DAh
9602 00003E83 B2DA             <1>       mov   dl, 0DAh
9603 00003E85 EC               <1>       in    al, dx
9604                            <1>
9605                            <1>         ; for(i=0;i<=8;i++) {
9606 00003E86 28C0             <1>       sub   al, al ; 0
9607                            <1>       ;;mov  dx, VGAREG_GRDC_ADDRESS ; 3CEh
9608                            <1>       ;mov   dl, 0CEh
9609 00003E88 B509             <1>       mov   ch, 9
9610                            <1> bfn_rvs_5:
9611                            <1>       ; outb(VGAREG_GRDC_ADDRESS,i)
```

```
9612                                    <1>     ;mov   dx, VGAREG_GRDC_ADDRESS ; 3CEh
9613 00003E8A B2CE                      <1>     mov    dl, 0CEh
9614 00003E8C EE                        <1>     out    dx, al
9615                                    <1>     ; outb(VGAREG_ACTL_READ_DATA)
9616 00003E8D 50                        <1>     push   eax
9617                                    <1>     ;mov   dx, VGAREG_GRDC_DATA ; 3CFh
9618                                    <1>     ;mov   dl, 0CFh
9619 00003E8E FEC2                      <1>     inc    dl
9620 00003E90 AC                        <1>     lodsb  ; (9 bytes in loop)
9621 00003E91 EE                        <1>     out    dx, al
9622 00003E92 58                        <1>     pop    eax
9623                                    <1>     ;dec   dl
9624 00003E93 FEC0                      <1>     inc    al  ; i++
9625 00003E95 FECD                      <1>     dec    ch
9626 00003E97 75F1                      <1>     jnz    short bfn_rvs_5
9627                                    <1>
9628                                    <1>     ; BX += 2; /* crtc_addr */     ; 3D4h
9629                                    <1>       ; BX += 4; /* plane latches */ ; 0
9630 00003E99 83C606                    <1>     add    esi, 6
9631 00003E9C 56                        <1>     push   esi ; *
9632                                    <1>
9633                                    <1>     ;outb(VGAREG_SEQU_ADDRESS, read_byte(ES, addr1)); addr1++;
9634                                    <1>       ;outb(crtc_addr, read_byte(ES, addr1)); addr1++;
9635                                    <1>       ;outb(VGAREG_GRDC_ADDRESS, read_byte(ES, addr1)); addr1++;
9636                                    <1>       ;addr1++;
9637                                    <1>       ;outb(crtc_addr - 0x4 + 0xa, read_byte(ES, addr1)); addr1++;
9638                                    <1>
9639 00003E9D 89FE                      <1>     mov    esi, edi ; start of state buffer
9640                                    <1>
9641                                    <1>     ;mov   dx, VGAREG_SEQU_ADDRESS ; 3C7h
9642 00003E9F B2C7                      <1>     mov    dl, 0C7h
9643 00003EA1 AC                        <1>     lodsb
9644 00003EA2 EE                        <1>     out    dx, al
9645                                    <1>     ;mov   dx, VGAREG_VGA_CRTC_ADDRESS ; 3D4h
9646 00003EA3 B2D4                      <1>     mov    dl, 0D4h
9647 00003EA5 AC                        <1>     lodsb
9648 00003EA6 EE                        <1>     out    dx, al
9649                                    <1>     ;mov   dx, VGAREG_GRDC_ADDRESS ; 3CEh
9650 00003EA7 B2CE                      <1>       mov  dl, 0CEh
9651 00003EA9 AC                        <1>     lodsb
9652 00003EAA EE                        <1>     out    dx, al
9653 00003EAB AC                        <1>     lodsb  ; addr1++
9654                                    <1>     ;mov   dx, VGAREG_VGA_WRITE_FEATURE_CTL ; 3DAh
9655 00003EAC B2DA                      <1>     mov    dl, 0DAh
9656 00003EAE AC                        <1>     lodsb
9657 00003EAF EE                        <1>     out    dx, al
9658                                    <1>
9659 00003EB0 5E                        <1>     pop    esi ; *
9660                                    <1>
9661                                    <1>     ; (total 70 bytes are read above as controller hardware state)
9662                                    <1>
9663                                    <1> bfn_rvs_6:
9664                                    <1>     ; 13/01/2021
9665 00003EB1 F6C102                    <1>     test   cl, 2
9666 00003EB4 747D                      <1>     jz     short bfn_rvs_9
9667                                    <1>
9668                                    <1>     ; VIDEO BIOS DATA
9669                                    <1>     ; !!! this data is valid for TRDOS 386 v2 kernel only !!!
9670                                    <1>     ; (this is not same with BOCHS/PLEX86 video bios, BIOS data)
9671                                    <1>
9672                                    <1>     ; if (CX & 2) {
9673                                    <1>     ;write_byte(BIOSMEM_SEG,BIOSMEM_CURRENT_MODE, read_byte(ES, BX)); BX++;
9674                                    <1>       ;write_word(BIOSMEM_SEG,BIOSMEM_NB_COLS, read_word(ES, BX)); BX += 2;
9675                                    <1>       ;write_word(BIOSMEM_SEG,BIOSMEM_PAGE_SIZE, read_word(ES, BX)); BX += 2;
9676                                    <1>       ;write_word(BIOSMEM_SEG,BIOSMEM_CRTC_ADDRESS, read_word(ES, BX)); BX += 2;
9677                                    <1>       ;write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, read_byte(ES, BX)); BX++;
9678                                    <1>       ;write_word(BIOSMEM_SEG,BIOSMEM_CHAR_HEIGHT, read_word(ES, BX)); BX += 2;
9679                                    <1>       ;write_byte(BIOSMEM_SEG,BIOSMEM_VIDEO_CTL, read_byte(ES, BX)); BX++;
9680                                    <1>       ;write_byte(BIOSMEM_SEG,BIOSMEM_SWITCHES, read_byte(ES, BX)); BX++;
9681                                    <1>       ;write_byte(BIOSMEM_SEG,BIOSMEM_MODESET_CTL, read_byte(ES, BX)); BX++;
9682                                    <1>       ;write_word(BIOSMEM_SEG,BIOSMEM_CURSOR_TYPE, read_word(ES, BX)); BX += 2;
9683                                    <1>       ;for(i=0;i<8;i++) {
9684                                    <1>       ;   write_word(BIOSMEM_SEG, BIOSMEM_CURSOR_POS+2*i, read_word(ES, BX));
9685                                    <1>       ;   BX += 2;
9686                                    <1>       ;}
9687                                    <1>       ;write_word(BIOSMEM_SEG,BIOSMEM_CURRENT_START, read_word(ES, BX)); BX += 2;
9688                                    <1>       ;write_byte(BIOSMEM_SEG,BIOSMEM_CURRENT_PAGE, read_byte(ES, BX)); BX++;
9689                                    <1>       ;/* current font */
9690                                    <1>       ;write_word(0, 0x1f * 4, read_word(ES, BX)); BX += 2;
9691                                    <1>       ;write_word(0, 0x1f * 4 + 2, read_word(ES, BX)); BX += 2;
9692                                    <1>       ;write_word(0, 0x43 * 4, read_word(ES, BX)); BX += 2;
9693                                    <1>       ;write_word(0, 0x43 * 4 + 2, read_word(ES, BX)); BX += 2;
9694                                    <1>
9695                                    <1>     ; !!! save TRDOS 386 v2 kernel spesific video bios data !!!
9696                                    <1>     ; (which is/are used by 'SET_MODE' function and/or it's sub functions)
9697                                    <1>
9698 00003EB6 66AD                      <1>     lodsw  ; CRTC_ADDR, always 3D4h (color VGA) for TRDOS 386 v2
9699                                    <1>     ; skip 3D4h check if it is already checked
9700 00003EB8 F6C101                    <1>     test   cl, 1
9701 00003EBB 7508                      <1>     jnz    short bfn_rvs_7
9702 00003EBD 663DD403                  <1>     cmp    ax, 3D4h
9703 00003EC1 7402                      <1>     je     short bfn_rvs_7
9704 00003EC3 F9                        <1>     stc
9705 00003EC4 C3                        <1>     retn
9706                                    <1> bfn_rvs_7:
9707 00003EC5 AC                        <1>     lodsb
9708 00003EC6 A2[BA6F0000]              <1>     mov    [CRT_MODE], al ; Current video mode (0FFh for VESA VBE modes)
9709 00003ECB AC                        <1>     lodsb
9710 00003ECC A2[BB6F0000]              <1>     mov    [CRT_MODE_SET], al ; 29h for mode 03h ; TRDOS 386 feature only !
9711 00003ED1 66AD                      <1>     lodsw
9712 00003ED3 66A3[1E120300]            <1>     mov    [video_mode], ax ; Current VESA VBE (SVGA, extended VGA) mode
9713 00003ED9 66AD                      <1>     lodsw          ; (valid if [CRT_MODE] = 0FFh)
9714 00003EDB 66A3[288E0100]            <1>     mov    [CRT_LEN], ax ; page size (in bytes)
9715 00003EE1 66AD                      <1>     lodsw
9716 00003EE3 66A3[AC810100]            <1>     mov    [CRT_START], ax ; video page start offset
```

```
9717 00003EE9 AC                  <1>       lodsb
9718 00003EEA A2[BC6F0000]        <1>       mov   [CRT_COLS], al ; nbcols, characters per row
9719 00003EEF AC                  <1>       lodsb
9720 00003EF0 A2[C26F0000]        <1>       mov   [VGA_ROWS], al ; nbrows, (character) rows per page (not rows-1)
9721 00003EF5 AC                  <1>       lodsb
9722 00003EF6 A2[BE6F0000]        <1>       mov   [CHAR_HEIGHT], al ; character font height (8 or 16 or 14)
9723 00003EFB AC                  <1>       lodsb
9724 00003EFC A2[BF6F0000]        <1>       mov   [VGA_VIDEO_CTL], al ; ROM BIOS DATA AREA Offset 87h
9725 00003F01 AC                  <1>       lodsb
9726 00003F02 A2[C06F0000]        <1>       mov   [VGA_SWITCHES], al ; feature bit switches
9727 00003F07 AC                  <1>       lodsb
9728 00003F08 A2[C16F0000]        <1>       mov   [VGA_MODESET_CTL], al ; basic mode set options
9729                              <1>       ; followings are only used by TRDOS 386 v2 (IBM PC/AT ROMBIOS) code
9730                              <1>       ; (bochs/plex86 does not use and return those)
9731 00003F0D AC                  <1>       lodsb
9732 00003F0E A2[BD6F0000]        <1>       mov   [CRT_PALETTE], al ; current color palette ; TRDOS 386 feature only !
9733 00003F13 AC                  <1>       lodsb
9734 00003F14 A2[BE810100]        <1>       mov   [ACTIVE_PAGE], al ; current video page
9735 00003F19 66AD                <1>       lodsw
9736 00003F1B 66A3[D36F0000]      <1>       mov   [CURSOR_MODE], ax ; cursor type
9737                              <1>       ;lodsd
9738                              <1>       ;mov  [CURSOR_POSN], eax ; cursor position for video page 0 and 1
9739                              <1>       ;lodsd
9740                              <1>       ;mov  [CURSOR_POSN+4], eax ; cursor position for video page 2 and 3
9741                              <1>       ;lodsd
9742                              <1>       ;mov  [CURSOR_POSN+8], eax ; cursor position for video page 4 and 5
9743                              <1>       ;lodsd
9744                              <1>       ;mov  [CURSOR_POSN+12], eax ; cursor position for video page 6 and 7
9745 00003F21 B504                <1>       mov   ch, 4
9746 00003F23 BF[AE810100]        <1>       mov   edi, CURSOR_POSN
9747                              <1> bfn_rvs_8:
9748 00003F28 A5                  <1>       movsd
9749 00003F29 FECD                <1>       dec   ch
9750 00003F2B 75FB                <1>       jnz   short bfn_rvs_8
9751                              <1>       ; (font addr) protected mode address in kernel's/system memory space
9752                              <1>       ; (not accessable/meaningful address value by user)
9753 00003F2D AD                  <1>       lodsd
9754 00003F2E A3[3A8E0100]        <1>       mov   [VGA_INT43H], eax ; VGA current (default) font address
9755                              <1>
9756                              <1>       ; (total 40 bytes are read&written above as BIOS data state)
9757                              <1> bfn_rvs_9:
9758                              <1>       ; 13/01/2021
9759 00003F33 F6C104              <1>       test  cl, 4
9760 00003F36 7422                <1>       jz    short bfn_rvs_11
9761                              <1>
9762                              <1>       ;BX++;
9763                              <1>         ;v = read_byte(ES, BX); BX++;
9764                              <1>         ;outb(VGAREG_PEL_MASK, read_byte(ES, BX)); BX++;
9765                              <1>         ;// Set the whole dac always, from 0
9766                              <1>         ;outb(VGAREG_DAC_WRITE_ADDRESS,0x00);
9767                              <1>         ;for(i=0;i<256*3;i++) {
9768                              <1>         ;   outb(VGAREG_DAC_DATA, read_byte(ES, BX)); BX++;
9769                              <1>         ;}
9770                              <1>         ;BX++;
9771                              <1>         ;outb(VGAREG_DAC_WRITE_ADDRESS, v);
9772                              <1>
9773                              <1>       ; /* read/write mode dac */
9774 00003F38 AC                  <1>       lodsb  ; skip ; VGAREG_DAC_STATE
9775 00003F39 AC                  <1>       lodsb
9776 00003F3A 88C4                <1>       mov   ah, al ; * ; v
9777 00003F3C AC                  <1>       lodsb
9778 00003F3D 66BAC603            <1>       mov   dx, VGAREG_PEL_MASK  ; 3C6h
9779 00003F41 EE                  <1>       out   dx, al
9780                              <1>       ;// Set the whole dac always, from 0
9781 00003F42 30C0                <1>       xor   al, al ; 0
9782                              <1>       ;mov  dx, VGAREG_DAC_WRITE_ADDRESS ; 3C8h
9783 00003F44 B2C8                <1>       mov   dl, 0C8h
9784 00003F46 EE                  <1>       out   dx, al
9785                              <1>
9786 00003F47 51                  <1>       push  ecx ; 22/01/2021
9787                              <1>       ;for(i=0;i<256*3;i++) {
9788 00003F48 B900030000          <1>       mov   ecx, 256*3 ; 768 bytes
9789                              <1>       ;mov  dx, VGAREG_DAC_DATA ; 3C9h
9790                              <1>       ;mov  dl, 0C9h
9791 00003F4D FEC2                <1>       inc   dl ; dx = 3C9h
9792                              <1> bfn_rvs_10:
9793 00003F4F AC                  <1>       lodsb
9794 00003F50 EE                  <1>       out   dx, al
9795 00003F51 E2FC                <1>       loop  bfn_rvs_10
9796 00003F53 59                  <1>       pop   ecx ; 22/01/2021
9797                              <1>
9798                              <1>       ; /* color select register */
9799 00003F54 AC                  <1>       lodsb  ; skip
9800                              <1>
9801 00003F55 88E0                <1>       mov   al, ah ; * ; v
9802                              <1>
9803                              <1>       ;mov  dx, VGAREG_DAC_WRITE_ADDRESS ; 3C8h
9804                              <1>       ;mov  dl, 0C8h
9805 00003F57 FECA                <1>       dec   dl ; dx  = 3C8h
9806 00003F59 EE                  <1>       out   dx, al ; * ; v
9807                              <1>
9808                              <1>       ; (total 772 bytes are read above as DAC state)
9809                              <1> bfn_rvs_11:
9810 00003F5A C3                  <1>       retn
9811                              <1>
9812                              <1> vbe_biosfn_restore_video_state:
9813                              <1>       ; 23/01/2021
9814                              <1>       ; 13/01/2021 (TRDOS 386 v2.0.3)
9815                              <1>       ; (vbe.c)
9816                              <1>
9817                              <1>       ; modified registers: eax, edx, esi, ch
9818                              <1>
9819                              <1>       ; input: esi = state buffer address
9820                              <1>       ; output:
9821                              <1>       ;       VBE DISPI register contents will be restored
```

```
                        <1>       ;       (18 bytes, 9 words)
                        <1>
                        <1>       ; enable = read_word(ES, BX);
                        <1>       ; BX += 2;
                        <1>       ;
                        <1>       ; if (!(enable & VBE_DISPI_ENABLED)) {
                        <1>         ;   outw(VBE_DISPI_IOPORT_INDEX,VBE_DISPI_INDEX_ENABLE);
                        <1>         ;   outw(VBE_DISPI_IOPORT_DATA, enable);
                        <1>       ; } else {
                        <1>         ;   outw(VBE_DISPI_IOPORT_INDEX, VBE_DISPI_INDEX_XRES);
                        <1>         ;   outw(VBE_DISPI_IOPORT_DATA, read_word(ES, BX));
                        <1>         ;   BX += 2;
                        <1>         ;   outw(VBE_DISPI_IOPORT_INDEX, VBE_DISPI_INDEX_YRES);
                        <1>         ;   outw(VBE_DISPI_IOPORT_DATA, read_word(ES, BX));
                        <1>         ;   BX += 2;
                        <1>         ;   outw(VBE_DISPI_IOPORT_INDEX, VBE_DISPI_INDEX_BPP);
                        <1>         ;   outw(VBE_DISPI_IOPORT_DATA, read_word(ES, BX));
                        <1>         ;   BX += 2;
                        <1>         ;   outw(VBE_DISPI_IOPORT_INDEX,VBE_DISPI_INDEX_ENABLE);
                        <1>         ;   outw(VBE_DISPI_IOPORT_DATA, enable);
                        <1>       ;
                        <1>         ;  for(i = VBE_DISPI_INDEX_BANK; i <= VBE_DISPI_INDEX_Y_OFFSET; i++)
                        <1>       ;    {
                        <1>         ;     outw(VBE_DISPI_IOPORT_INDEX, i);
                        <1>         ;     outw(VBE_DISPI_IOPORT_DATA, read_word(ES, BX));
                        <1>         ;     BX += 2;
                        <1>         ;    }
                        <1>         ; }
                        <1>
00003F5B 66AD           <1>       lodsw  ; enable (status, enabled=1, disabled=0)
00003F5D 66BACE01       <1>       mov    dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
                        <1>       ; 23/01/2021
00003F61 6683E001       <1>       and    ax, 1 ; VBE_DISPI_ENABLED
00003F65 750B           <1>       jnz    short vbe_bfn_rvs_1
                        <1>       ; ax = 0
                        <1>       ; VBE_DISPI_DISABLED
                        <1> vbe_bfn_rvs_0:
                        <1>       ; enable (disable) dispi
                        <1>       ; dx = 01CEh ; VBE_DISPI_IOPORT_INDEX
                        <1>       ; ah = 0
00003F67 50             <1>       push   eax
00003F68 B004           <1>       mov    al, 04h      ; VBE_DISPI_INDEX_ENABLE
00003F6A 66EF           <1>       out    dx, ax
                        <1>       ;mov   dx, 01CFh ; VBE_DISPI_IOPORT_DATA
00003F6C FEC2           <1>       inc    dl
00003F6E 58             <1>       pop    eax
00003F6F 66EF           <1>       out    dx, ax ; enable (or disable)
00003F71 C3             <1>       retn
                        <1> vbe_bfn_rvs_1:
                        <1>       ; VBE_DISPI_ENABLED
                        <1>
                        <1>       ; from VBE_DISPI_INDEX_XRES
                        <1>       ;   to VBE_DISPI_INDEX_BPP
                        <1>
00003F72 B503           <1>       mov    ch, 3
00003F74 28C0           <1>       sub    al, al ; 0 ; VBE_DISPI_INDEX_XRES - 1
                        <1>       ; ax = 0
                        <1>
00003F76 E80B000000     <1>       call   vbe_bfn_rvs_2
                        <1>
                        <1>         ;outw(VBE_DISPI_IOPORT_INDEX,VBE_DISPI_INDEX_ENABLE);
                        <1>       ;outw(VBE_DISPI_IOPORT_DATA, enable);
                        <1>
                        <1>       ; 23/01/2021
00003F7B B001           <1>       mov    al, 1 ; VBE_DISPI_ENABLED
                        <1>       ; ax = 1
00003F7D E8E5FFFFFF     <1>       call   vbe_bfn_rvs_0
                        <1>
                        <1>       ; from VBE_DISPI_INDEX_BANK
                        <1>       ;   to VBE_DISPI_INDEX_Y_OFFSET
                        <1>
00003F82 B505           <1>       mov    ch, 5
                        <1>       ; 23/01/2021
00003F84 B004           <1>       mov    al, 4  ; VBE_DISPI_INDEX_BANK - 1
                        <1>       ; ax = 4
                        <1> vbe_bfn_rvs_2:
00003F86 FEC0           <1>       inc    al ; from VBE_DISPI_INDEX_XRES
                        <1>                 ; to VBE_DISPI_INDEX_BPP
                        <1>       ;mov   dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
                        <1>       ;mov   dl, 0CEh
00003F88 66EF           <1>       out    dx, ax
00003F8A 50             <1>       push   eax
                        <1>       ;mov   dx, 01CFh ; VBE_DISPI_IOPORT_DATA
00003F8B FEC2           <1>       inc    dl ; 1CFh
00003F8D 66AD           <1>       lodsw
00003F8F 66EF           <1>       out    dx, ax
00003F91 58             <1>       pop    eax
00003F92 FECA           <1>       dec    dl ; 1CEh
00003F94 FECD           <1>       dec    ch
00003F96 75EE           <1>       jnz    short vbe_bfn_rvs_2
00003F98 C3             <1>       retn
                        <1>
                        <1> ; ---------------------------------------------------------
                        <1>
                        <1> dispi_set_enable:
                        <1>       ; 23/11/2020
                        <1>       ; Input:
                        <1>       ;     ax = VBE_DISPI_ENABLED = 1
                        <1>       ;          or VBE_DISPI_DISABLED = 0
                        <1>       ;
                        <1>       ; Modified registers: none
                        <1>
                        <1>       ;push  edx
                        <1>       ;push  eax
                        <1>       ;mov   dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
```

```
                                <1>       ;mov   ax, 04h        ; VBE_DISPI_INDEX_ENABLE
                                <1>       ;out   dx, ax
                                <1>       ;pop   eax
                                <1>       ;;mov  dx, 01CFh ; VBE_DISPI_IOPORT_DATA
                                <1>       ;;mov  dl, 0CFh
                                <1>       ;inc   dl
                                <1>       ;out   dx, ax
                                <1>       ;pop   edx
                                <1>       ;retn
                                <1>
                                <1>       ; 25/11/2020
                                <1>       ; Modified registers: edx
                                <1>       ;push  edx
00003F99 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>
00003F9D 50                     <1>       push   eax
00003F9E 6689D0                 <1>       mov    ax, dx
00003FA1 66BACE01               <1>       mov    dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
00003FA5 66EF                   <1>       out    dx, ax
00003FA7 58                     <1>       pop    eax
                                <1>       ;mov   dx, 01CFh ; VBE_DISPI_IOPORT_DATA
                                <1>       ;mov   dl, 0CFh
00003FA8 FEC2                   <1>       inc    dl
00003FAA 66EF                   <1>       out    dx, ax
00003FAC 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
00003FAD 66BA0300               <1>       mov    dx, 03h ; VBE_DISPI_INDEX_BPP
                                <1>       ;call  dispi_set_parms
                                <1>       ;pop   edx
                                <1>       ;retn
00003FB1 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
                                <1>
                                <1>       ; 25/11/2020
                                <1>       ; Modified registers: edx
                                <1>       ;push  edx
00003FB3 66BA0100               <1>       mov    dx, 01h ; VBE_DISPI_INDEX_XRES
                                <1>       ;call  dispi_set_parms
                                <1>       ;pop   edx
                                <1>       ;retn
00003FB7 EBE4                   <1>       jmp    short dispi_set_parms
                                <1>
                                <1> dispi_set_yres:
                                <1>       ; 25/11/2020
                                <1>       ; Input:
                                <1>       ;      ax = Y resolution (screen height)
                                <1>       ;           (200,400,600,720,768,1080)
                                <1>       ;
                                <1>       ; Modified registers: none
```

```
                                <1>
                                <1>       ;push  edx
                                <1>       ;push  eax
                                <1>       ;mov   dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
                                <1>       ;mov   ax, 02h        ; VBE_DISPI_INDEX_YRES
                                <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
00003FB9 66BA0200               <1>       mov    dx, 02h  ; VBE_DISPI_INDEX_YRES
                                <1>       ;call  dispi_set_parms
                                <1>       ;pop   edx
                                <1>       ;retn
00003FBD EBDE                   <1>       jmp    short dispi_set_parms
                                <1>
                                <1> dispi_set_bank:
                                <1>       ; 25/11/2020
                                <1>       ; Input:
                                <1>       ;      ax = video memory bank number
                                <1>       ;
                                <1>       ; Modified registers: none
                                <1>
                                <1>       ;push  edx
                                <1>       ;push  eax
                                <1>       ;mov   dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
                                <1>       ;mov   ax, 05h        ; VBE_DISPI_INDEX_BANK
                                <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 66BA0500               <1>       mov    dx, 05h  ; VBE_DISPI_INDEX_BANK
                                <1>       ;call  dispi_set_parms
                                <1>       ;pop   edx
                                <1>       ;retn
00003FC3 EBD8                   <1>       jmp    short dispi_set_parms
                                <1>
                                <1> dispi_get_enable:
                                <1>       ; 27/11/2020
                                <1>       ; Input:
                                <1>       ;      none
                                <1>       ; Output:
                                <1>       ;      ax = vbe dispi status
                                <1>       ;
                                <1>       ; Modified registers: eax
                                <1>
                                <1>       ;push  edx
                                <1>       ;mov   dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
                                <1>       ;mov   ax, 04h        ; VBE_DISPI_INDEX_ENABLE
                                <1>       ;out   dx, ax
                                <1>       ;;mov  dx, 01CFh ; VBE_DISPI_IOPORT_DATA
                                <1>       ;;mov  dl, 0CFh
                                <1>       ;inc   dl
                                <1>       ;in    ax, dx
                                <1>       ;pop   edx
                                <1>       ;retn
                                <1>
                                <1>       ; 27/11/2020
                                <1>       ; Modified registers: eax, edx
                                <1>       ;push  edx
00003FC5 66B80400               <1>       mov    ax, 04h ; VBE_DISPI_INDEX_ENABLE
                                <1>       ;call  dispi_get_parms
                                <1>       ;pop   edx
                                <1>       ;retn
                                <1>       ;;jmp  short dispi_get_parms
                                <1>
                                <1> dispi_get_parms:
                                <1>       ; 25/11/2020
                                <1>       ; Input:
                                <1>       ;      ax = vbe dispi register index
                                <1>       ; output:
                                <1>       ;      ax = data
                                <1>       ;
                                <1>       ; Modified registers: eax, edx
                                <1>
00003FC9 66BACE01               <1>       mov    dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
00003FCD 66EF                   <1>       out    dx, ax
                                <1>       ;mov   dx, 01CFh ; VBE_DISPI_IOPORT_DATA
                                <1>       ;mov   dl, 0CFh
00003FCF FEC2                   <1>       inc    dl
00003FD1 66ED                   <1>       in     ax, dx
00003FD3 C3                     <1>       retn
                                <1>
                                <1> vga_compat_setup:
                                <1>       ; 26/11/2020
                                <1>       ; 25/11/2020
                                <1>       ; VGA compatibility setup
                                <1>       ; (vbe.c, 02/01/2020, vruppert)
                                <1>       ;
                                <1>       ; Input:
```

```
10137                                  <1>       ;      none
10138                                  <1>       ;
10139                                  <1>       ; Modified registers: eax, edx
10140                                  <1>
10141                                  <1>       ; 26/11/2020
10142                                  <1>       ;push  eax
10143                                  <1>       ;push  edx
10144                                  <1>
10145                                  <1>       ; set CRT X resolution
10146 00003FD4 66BACE01               <1>       mov    dx, 1CEh ; VBE_DISPI_IOPORT_INDEX
10147 00003FD8 66B80100               <1>       mov    ax, 01h  ; VBE_DISPI_INDEX_XRES
10148 00003FDC 66EF                   <1>       out    dx, ax
10149                                  <1>       ;mov   dx, 1CFh ; VBE_DISPI_IOPORT_DATA
10150 00003FDE FEC2                   <1>       inc    dl
10151 00003FE0 66ED                   <1>       in     ax, dx
10152 00003FE2 50                     <1>       push   eax
10153 00003FE3 66BAD403               <1>       mov    dx, 3D4h ; VGAREG_VGA_CRTC_ADDRESS
10154 00003FE7 66B81100               <1>       mov    ax, 0011h ; Vertical retrace end register
10155 00003FEB 66EF                   <1>       out    dx, ax
10156                                  <1>       ;pop   eax
10157                                  <1>       ;push  eax
10158 00003FED 8B0424                 <1>       mov    eax, [esp]
10159 00003FF0 66C1E803               <1>       shr    ax, 3 ; / 8 for pixel to character
10160 00003FF4 6648                   <1>       dec    ax  ; - 1 (EGA or VGA?)
10161 00003FF6 88C4                   <1>       mov    ah, al
10162 00003FF8 B001                   <1>       mov    al, 01h ; Horizontal display end register
10163 00003FFA 66EF                   <1>       out    dx, ax
10164 00003FFC 58                     <1>       pop    eax
10165                                  <1>
10166 00003FFD E8B0000000             <1>       call   vga_set_virt_width
10167                                  <1>
10168                                  <1>       ; set CRT Y resolution
10169 00004002 66BACE01               <1>       mov    dx, 1CEh ; VBE_DISPI_IOPORT_INDEX
10170 00004006 66B80200               <1>       mov    ax, 02h  ; VBE_DISPI_INDEX_YRES
10171 0000400A 66EF                   <1>       out    dx, ax
10172                                  <1>       ;mov   dx, 1CFh ; VBE_DISPI_IOPORT_DATA
10173 0000400C FEC2                   <1>       inc    dl
10174 0000400E 66ED                   <1>       in     ax, dx
10175 00004010 50                     <1>       push   eax
10176 00004011 66BAD403               <1>       mov    dx, 3D4h ; VGAREG_VGA_CRTC_ADDRESS
10177 00004015 88C4                   <1>       mov    ah, al
10178 00004017 B012                   <1>       mov    al, 12h ; Vertical display end register
10179 00004019 66EF                   <1>       out    dx, ax
10180 0000401B 58                     <1>       pop    eax
10181 0000401C B007                   <1>       mov    al, 07h      ; Overflow register
10182 0000401E EE                     <1>       out    dx, al
10183 0000401F 6642                   <1>       inc    dx
10184 00004021 EC                     <1>       in     al, dx ; read overflow register
10185 00004022 24BD                   <1>       and    al, 0BDh ; clear VDE 9th and 10th bits
10186 00004024 F6C401                 <1>       test   ah, 01h
10187 00004027 7402                   <1>       jz     short bit8_clear
10188 00004029 0C02                   <1>       or     al, 02h ; VDE 9th bit (bit 8) in bit 1
10189                                  <1> bit8_clear:
10190 0000402B F6C402                 <1>       test   ah, 02h
10191 0000402E 7402                   <1>       jz     short bit9_clear
10192 00004030 0C40                   <1>       or     al, 40h ; VDE 10th bit (bit 9) in bit 6
10193                                  <1> bit9_clear:
10194 00004032 EE                     <1>       out    dx, al
10195                                  <1>
10196                                  <1>       ; other settings
10197 00004033 66BAD403               <1>       mov    dx, 3D4h ; VGAREG_VGA_CRTC_ADDRESS
10198 00004037 66B80900               <1>       mov    ax, 0009h ; Maximum scan line register
10199 0000403B 66EF                   <1>       out    dx, ax ; Reset
10200 0000403D B017                   <1>       mov    al, 17h ; Mode control register
10201 0000403F EE                     <1>       out    dx, al
10202                                  <1>       ;mov   dx, 3D5h ; VGAREG_VGA_CRTC_DATA
10203 00004040 FEC2                   <1>       inc    dl
10204 00004042 EC                     <1>       in     al, dx ; Read mode control register
10205 00004043 0C03                   <1>       or     al, 03h ; Set SRS and CMS bits
10206 00004045 EE                     <1>       out    dx, al
10207 00004046 66BADA03               <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
10208 0000404A EC                     <1>       in     al, dx  ; clear flip-flop
10209 0000404B 66BAC003               <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
10210 0000404F B010                   <1>       mov    al, 10h      ; Mode control register
10211 00004051 EE                     <1>       out    dx, al
10212                                  <1>       ;mov   dx, 3C1h ; VGAREG_ACTL_READ_DATA
10213 00004052 FEC2                   <1>       inc    dl
10214 00004054 EC                     <1>       in     al, dx
10215 00004055 0C01                   <1>       or     al, 01h ; select graphics mode
10216                                  <1>       ;mov   dx, 3C0h ; VGAREG_ACTL_ADDRESS
10217 00004057 FECA                   <1>       dec    dl
10218 00004059 EE                     <1>       out    dx, al ; Write to mode control register
10219 0000405A B020                   <1>       mov    al, 20h ; Palette RAM <-> display memory
10220 0000405C EE                     <1>       out    dx, al ; Write to attribute addr register
10221 0000405D 66BACE03               <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
10222 00004061 66B80605               <1>       mov    ax, 0506h ; Misc. register, graph, mm 1
10223 00004065 66EF                   <1>       out    dx, ax
10224 00004067 66BAC403               <1>       mov    dx, 3C4h ; VGAREG_SEQU_ADDRESS
10225 0000406B 66B8020F               <1>       mov    ax, 0F02h ; Map mask register, all planes
10226 0000406F 66EF                   <1>       out    dx, ax
10227                                  <1>
10228                                  <1>       ; settings for >= 8bpp
10229                                  <1>
10230                                  <1>       ;mov   dx, 1CEh ; VBE_DISPI_IOPORT_INDEX
10231                                  <1>       ;mov   ax, 03h ; VBE_DISPI_INDEX_BPP
10232                                  <1>       ;out   dx, ax
10233                                  <1>       ;;mov  dx, 1CFh ; VBE_DISPI_IOPORT_DATA
10234                                  <1>       ;inc   dl
10235                                  <1>       ;in    ax, dx
10236                                  <1>       ;cmp   al, 08h  ; < 8 bits per pixel
10237                                  <1>       ;jb    short vga_compat_end
10238                                  <1>
10239 00004071 66BAD403               <1>       mov    dx, 3D4h ; VGAREG_VGA_CRTC_ADDRESS
10240 00004075 B014                   <1>       mov    al, 14h  ; Underline location register
10241 00004077 EE                     <1>       out    dx, al
```

```
10242                                  <1>       ;mov   dx, 3D5h ; VGAREG_VGA_CRTC_DATA
10243 00004078 FEC2                    <1>       inc    dl
10244 0000407A EC                      <1>       in     al, dx
10245 0000407B 0C40                    <1>       or     al, 40h       ; enable double word mode
10246 0000407D EE                      <1>       out    dx, al
10247 0000407E 66BADA03                <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
10248 00004082 EC                      <1>       in     al, dx  ; clear flip-flop
10249 00004083 66BAC003                <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
10250 00004087 B010                    <1>       mov    al, 10h       ; Mode control register
10251 00004089 EE                      <1>       out    dx, al
10252                                  <1>       ;mov   dx, 3C1h ; VGAREG_ACTL_READ_DATA
10253 0000408A FEC2                    <1>       inc    dl
10254 0000408C EC                      <1>       in     al, dx
10255 0000408D 0C40                    <1>       or     al, 40h  ; Pixel clock select is 1
10256                                  <1>       ;mov   dx, 3C0h ; VGAREG_ACTL_ADDRESS
10257 0000408F FECA                    <1>       dec    dl
10258 00004091 EE                      <1>       out    dx, al  ; update mode control reggister
10259 00004092 B020                    <1>       mov    al, 20h       ; select display memory as PAS
10260 00004094 EE                      <1>       out    dx, al
10261 00004095 66BAC403                <1>       mov    dx, 3C4h ; VGAREG_SEQU_ADDRESS
10262 00004099 B004                    <1>       mov    al, 04h      ; Memory mode register
10263 0000409B EE                      <1>       out    dx, al
10264                                  <1>       ;mov   dx, 3C5h ; VGAREG_SEQU_DATA
10265 0000409C FEC2                    <1>       inc    dl
10266 0000409E EC                      <1>       in     al, dx
10267 0000409F 0C08                    <1>       or     al, 08h      ; enable chain four
10268 000040A1 EE                      <1>       out    dx, al
10269 000040A2 66BACE03                <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
10270 000040A6 B005                    <1>       mov    al, 05h       ; Mode register
10271 000040A8 EE                      <1>       out    dx, al
10272                                  <1>       ;mov   dx, 3CFh ; VGAREG_GRDC_DATA
10273 000040A9 FEC2                    <1>       inc    dl
10274 000040AB EC                      <1>       in     al, dx
10275 000040AC 249F                    <1>       and    al, 9Fh       ; clear shift register
10276 000040AE 0C40                    <1>       or     al, 40h  ; set shift register to 2
10277 000040B0 EE                      <1>       out    dx, al
10278                                  <1>
10279                                  <1> vga_compat_end:
10280                                  <1>       ;pop   edx
10281                                  <1>       ;pop   eax
10282 000040B1 C3                      <1>       retn
10283                                  <1>
10284                                  <1> vga_set_virt_width:
10285                                  <1>       ; 27/11/2020
10286                                  <1>       ; 25/11/2020
10287                                  <1>       ; (vbe.c, 02/01/2020, vruppert)
10288                                  <1>       ;
10289                                  <1>       ; Input:
10290                                  <1>       ;      AX = resolution (screen width)
10291                                  <1>       ;
10292                                  <1>       ; Modified registers: eax, edx
10293                                  <1>
10294                                  <1>       ;;push ebx
10295                                  <1>       ;push  edx
10296                                  <1>       ;push  eax
10297                                  <1>       ;mov   ebx, eax
10298                                  <1>       ;call  dispi_get_bpp ; bits per pixel
10299                                  <1>       ;cmp   al, 4
10300                                  <1>       ;ja    short set_width_svga  ; 8, 16, 24, 32
10301                                  <1>       ;shr   bx, 1
10302                                  <1> ;set_width_svga:
10303                                  <1>       ;shr   bx, 3
10304                                  <1>       ;mov   eax, [esp]
10305 000040B2 66C1E803                <1>       shr    ax, 3  ; / 8, bytes per row
10306 000040B6 66BAD403                <1>       mov    dx, 3D4h ; VGAREG_VGA_CRTC_ADDRESS
10307                                  <1>       ;mov   ah, bl ;
10308 000040BA 88C4                    <1>       mov    ah, al ; width in bytes
10309 000040BC B013                    <1>       mov    al, 13h      ; offset register
10310 000040BE 66EF                    <1>       out    dx, ax ; index (3D4h) and data (3D5h)
10311                                  <1>       ;pop   eax
10312                                  <1>       ;pop   edx
10313                                  <1>       ;;pop  ebx
10314 000040C0 C3                      <1>       retn
10315                                  <1>
10316                                  <1> ; 24/11/2020
10317                                  <1>
10318                                  <1> struc bmi ; BOCHS/PLEX86 MODE INFO structure/table
10319 00000000 <res 00000002>          <1>  .mode:       resw 1
10320 00000002 <res 00000002>          <1>  .width:  resw 1
10321 00000004 <res 00000002>          <1>  .height: resw 1
10322 00000006 <res 00000002>          <1>  .depth:  resw 1
10323                                  <1>  .size:
10324                                  <1> endstruc
10325                                  <1>
10326                                  <1> ; 24/11/2020
10327                                  <1> struc MODEINFO
10328 00000000 <res 00000002>          <1>  .mode:                   resw 1  ; 1XXh
10329 00000002 <res 00000002>          <1>  .ModeAttributes:  resw 1
10330 00000004 <res 00000001>          <1>  .WinAAttributes:  resb 1
10331 00000005 <res 00000001>          <1>  .WinBAttributes:  resb 1 ; = 0
10332 00000006 <res 00000002>          <1>  .WinGranularity:  resw 1
10333 00000008 <res 00000002>          <1>  .WinSize:         resw 1
10334 0000000A <res 00000002>          <1>  .WinASegment:            resw 1
10335 0000000C <res 00000002>          <1>  .WinBSegment:            resw 1 ; = 0
10336 0000000E <res 00000004>          <1>  .WinFuncPtr:             resd 1 ; = 0
10337 00000012 <res 00000002>          <1>  .BytesPerScanLine:       resw 1
10338 00000014 <res 00000002>          <1>  .XResolution:            resw 1
10339 00000016 <res 00000002>          <1>  .YResolution:            resw 1
10340 00000018 <res 00000001>          <1>  .XCharSize:       resb 1
10341 00000019 <res 00000001>          <1>  .YCharSize:       resb 1
10342 0000001A <res 00000001>          <1>  .NumberOfPlanes:  resb 1
10343 0000001B <res 00000001>          <1>  .BitsPerPixel:           resb 1
10344 0000001C <res 00000001>          <1>  .NumberOfBanks:   resb 1
10345 0000001D <res 00000001>          <1>  .MemoryModel:            resb 1
10346 0000001E <res 00000001>          <1>  .BankSize:        resb 1 ; = 0
```

```
10347 0000001F <res 00000001>      <1>  .NumberOfImagePages:     resb 1
10348 00000020 <res 00000001>      <1>  .Reserved_page:   resb 1 ; = 0
10349 00000021 <res 00000001>      <1>  .RedMaskSize:            resb 1
10350 00000022 <res 00000001>      <1>  .RedFieldPosition:       resb 1
10351 00000023 <res 00000001>      <1>  .GreenMaskSize:   resb 1
10352 00000024 <res 00000001>      <1>  .GreenFieldPosition:     resb 1
10353 00000025 <res 00000001>      <1>  .BlueMaskSize:           resb 1
10354 00000026 <res 00000001>      <1>  .BlueFieldPosition:      resb 1
10355 00000027 <res 00000001>      <1>  .RsvdMaskSize:           resb 1
10356 00000028 <res 00000001>      <1>  .RsvdFieldPosition:      resb 1
10357 00000029 <res 00000001>      <1>  .DirectColorModeInfo:    resb 1
10358 0000002A <res 00000004>      <1>  .PhysBasePtr:            resd 1
10359 0000002E <res 00000004>      <1>  .OffScreenMemOffset:     resd 1 ; = 0
10360 00000032 <res 00000002>      <1>  .OffScreenMemSize:       resw 1 ; = 0
10361 00000034 <res 00000002>      <1>  .LinBytesPerScanLine:    resw 1
10362 00000036 <res 00000001>      <1>  .BnkNumberOfPages:       resb 1
10363 00000037 <res 00000001>      <1>  .LinNumberOfPages:       resb 1
10364 00000038 <res 00000001>      <1>  .LinRedMaskSize:  resb 1
10365 00000039 <res 00000001>      <1>  .LinRedFieldPosition1: resb 1
10366 0000003A <res 00000001>      <1>  .LinGreenMaskSize1:      resb 1
10367 0000003B <res 00000001>      <1>  .LinGreenFieldPosition:resb 1
10368 0000003C <res 00000001>      <1>  .LinBlueMaskSize:        resb 1
10369 0000003D <res 00000001>      <1>  .LinBlueFieldPosition:   resb 1
10370 0000003E <res 00000001>      <1>  .LinRsvdMaskSize:        resb 1
10371 0000003F <res 00000001>      <1>  .LinRsvdFieldPosition:   resb 1
10372 00000040 <res 00000004>      <1>  .MaxPixelClock:   resd 1 ; = 0
10373                              <1> .size:
10374                              <1> endstruc
10375                              <1>
10376                              <1> ; 10/12/2020
10377                              <1> struc LFBINFO
10378 00000000 <res 00000002>      <1>  .mode:                   resw 1  ; 1XXh
10379 00000002 <res 00000004>      <1>  .LFB_addr:        resd 1
10380 00000006 <res 00000004>      <1>  .LFB_size:        resd 1
10381 0000000A <res 00000002>      <1>  .X_res:           resw 1
10382 0000000C <res 00000002>      <1>  .Y_res:           resw 1
10383 0000000E <res 00000001>      <1>  .bpp:                    resb 1
10384 0000000F <res 00000001>      <1>  .reserved:        resb 1
10385                              <1> .size:      ; 16 bytes
10386                              <1> endstruc
10387                              <1>
10388                              <1> set_mode_info_list:
10389                              <1>       ; 14/12/2020
10390                              <1>       ; 11/12/2020
10391                              <1>       ; 24/11/2020
10392                              <1>       ; (vbetables-gen.c)
10393                              <1>       ; Input:
10394                              <1>       ;     BX = VBE mode (including bochs special modes)
10395                              <1>       ; Output:
10396                              <1>       ;     ;;EAX = MODE_INFO_LIST address
10397                              <1>       ;     EAX = 0 ; 11/12/2020
10398                              <1>       ;     ESI = MODE_INFO_LIST address ; 11/12/2020
10399                              <1>       ;     (if mode is not found, ESI = 0)
10400                              <1>       ;
10401                              <1>       ; Modified registers: eax, ebx, ecx, edx, esi, edi
10402                              <1>
10403 000040C1 BE[46730000]       <1>       mov   esi, b_vbe_modes ; bochs mode info base table
10404 000040C6 BF[3A120300]       <1>       mov   edi, MODE_INFO_LIST ; mode info list (4F01h)
10405                              <1> sml_0:
10406 000040CB 66AD               <1>       lodsw
10407 000040CD 6639D8             <1>       cmp   ax, bx ; is mode number same ?
10408 000040D0 7410               <1>       je    short sml_1 ; yes
10409 000040D2 AD                 <1>       lodsd
10410 000040D3 66AD               <1>       lodsw
10411 000040D5 81FE[06740000]     <1>       cmp   esi, end_of_b_vbe_modes
10412 000040DB 72EE               <1>       jb    short sml_0
10413                              <1>       ; not found
10414 000040DD 31C0               <1>       xor   eax, eax ; 0
10415                              <1>       ; 11/12/2020
10416 000040DF 31F6               <1>       xor   esi, esi
10417 000040E1 C3                 <1>       retn
10418                              <1> sml_1:
10419 000040E2 66AB               <1>       stosw ; mode
10420 000040E4 AD                 <1>       lodsd ; width, height
10421                              <1>       ; 14/12/2020
10422 000040E5 89C1               <1>       mov   ecx, eax
10423 000040E7 50                 <1>       push  eax ; ***
10424 000040E8 29C0               <1>       sub   eax, eax ; clear high word of eax
10425 000040EA 66AD               <1>       lodsw ; depth
10426 000040EC 50                 <1>       push  eax ; **
10427                              <1>
10428                              <1>       ;add  al, 7 ; only for 15 bit colors (not used here)
10429 000040ED C0E803             <1>       shr   al, 3 ; / 8
10430                              <1>       ; 14/12/2020
10431 000040F0 66F7E1             <1>       mul   cx  ; pitch = width * ((depth+7)/8)
10432                              <1>             ; ax = pitch
10433 000040F3 50                 <1>       push  eax ; * ; high word of eax = 0
10434 000040F4 C1E910             <1>       shr   ecx, 16
10435                              <1>       ;mul  cx
10436                              <1>       ;mov  cx, ax
10437 000040F7 31D2               <1>       xor   edx, edx  ; clear high word of edx
10438 000040F9 F7E1               <1>       mul   ecx ; height * pitch
10439 000040FB 89C1               <1>       mov   ecx, eax
10440 000040FD B800000001         <1>       mov   eax, VBE_DISPI_TOTAL_VIDEO_MEMORY_MB * 1024 * 1024
10441 00004102 F7F1               <1>       div   ecx
10442                              <1>             ; eax = pages = vram_size / (height*pitch)
10443                              <1>
10444                              <1>       ;mov  cx, ax
10445 00004104 89C1               <1>       mov   ecx, eax ; pages
10446                              <1>
10447 00004106 66B89B00           <1>       mov   ax, MODE_ATTRIBUTES
10448 0000410A 66AB               <1>       stosw ; ModeAttributes
10449 0000410C B007               <1>       mov   al, WINA_ATTRIBUTES
10450 0000410E AA                 <1>       stosb ; WinAAttributes
10451 0000410F 30C0               <1>       xor   al, al ; WinBAttributes = 0
```

```
10452 00004111 AA                <1>       stosb
10453 00004112 66B84000          <1>       mov    ax, VBE_DISPI_BANK_SIZE_KB
10454 00004116 66AB              <1>       stosw  ; WinGranularity
10455 00004118 66AB              <1>       stosw  ; WinSize
10456 0000411A 66B800A0          <1>       mov    ax, VGAMEM_GRAPH
10457 0000411E 66AB              <1>       stosw  ; WinASegment
10458 00004120 29C0              <1>       sub    eax, eax
10459 00004122 66AB              <1>       stosw  ; WinBSegment = 0
10460 00004124 AB                <1>       stosd  ; WinFuncPtr = 0
10461                            <1>
10462 00004125 58                <1>       pop    eax ; * ; pitch
10463 00004126 89C3              <1>       mov    ebx, eax  ; high word of ebx = 0 ; 14/12/2020
10464 00004128 66AB              <1>       stosw  ; BytesPerScanLine
10465                            <1>
10466 0000412A 5A                <1>       pop    edx ; ** ; depth (bits per pixel)
10467 0000412B 58                <1>       pop    eax ; *** width, height
10468                            <1>
10469                            <1>       ; // Mandatory information for VBE 1.2 and above
10470                            <1>
10471 0000412C 66AB              <1>       stosw  ; XResolution (width)
10472 0000412E C1E810            <1>       shr    eax, 16
10473 00004131 50                <1>       push   eax ; **** height
10474 00004132 66AB              <1>       stosw  ; YResolution (height)
10475 00004134 B008              <1>       mov    al, 8
10476 00004136 AA                <1>       stosb  ; XCharSize  ; char width
10477 00004137 B010              <1>       mov    al, 16
10478 00004139 AA                <1>       stosb  ; YCharSize  ; char height
10479 0000413A B001              <1>       mov    al, 1
10480 0000413C AA                <1>       stosb  ; NumberOfPlanes
10481                            <1>       ;movzx eax, dl
10482 0000413D 88D0              <1>       mov    al, dl ; eax <= 32
10483 0000413F AA                <1>       stosb  ; BitsPerPixel
10484                            <1>       ; Number of banks = (height * pitch + 65535) / 65536
10485 00004140 58                <1>       pop    eax ; **** ; height
10486                            <1>       ; 14/12/2020
10487 00004141 52                <1>       push   edx ; ***** ; depth ; edx <= 32
10488 00004142 F7E3              <1>       mul    ebx  ; pitch (ebx) * height (eax)
10489                            <1>       ;mov   edx, [esp] ; *****
10490                            <1>       ;mov   dl, [esp] ; *****
10491 00004144 05FFFF0000        <1>       add    eax, 65535
10492 00004149 C1E810            <1>       shr    eax, 16 ; / 65536 ; <= 127 ; 14/12/2020
10493 0000414C AA                <1>       stosb  ; NumberOfBanks
10494                            <1>       ; 14/12/2020
10495                            <1>       ;cmp   dl, 8 ; 8 bits per pixel
10496 0000414D 803C2408          <1>       cmp    byte [esp], 8
10497 00004151 7704              <1>       ja     short sml_2
10498 00004153 B004              <1>       mov    al, VBE_MEMORYMODEL_PACKED_PIXEL
10499 00004155 EB02              <1>       jmp    short sml_3
10500                            <1> sml_2:
10501                            <1>       ; 16, 24, 32 bts per pixel
10502 00004157 B006              <1>       mov    al, VBE_MEMORYMODEL_DIRECT_COLOR
10503                            <1> sml_3:
10504 00004159 AA                <1>       stosb
10505 0000415A 30C0              <1>       xor    al, al ; 0
10506 0000415C AA                <1>       stosb  ; BankSize = 0
10507 0000415D 49                <1>       dec    ecx ; pages - 1
10508                            <1>       ; NumberOfImagePages = 262 for 320x200x8 mode
10509                            <1>       ;;mov  ax, 255
10510                            <1>       ; 14/12/2020
10511                            <1>       ;mov   al, 255
10512 0000415E FEC8              <1>       dec    al ; 255
10513 00004160 39C1              <1>       cmp    ecx, eax ; ecx <= 261, eax = 255
10514                            <1>       ;cmp   cx, ax
10515 00004162 7302              <1>       jnb    short sml_4
10516 00004164 88C8              <1>       mov    al, cl
10517                            <1> sml_4:
10518 00004166 AA                <1>       stosb  ; NumberOfImagePages (1 byte)
10519 00004167 28C0              <1>       sub    al, al
10520 00004169 AA                <1>       stosb  ; Reserved_page = 0
10521 0000416A 58                <1>       pop    eax ; ***** ; depth
10522 0000416B 88C1              <1>       mov    cl, al
10523                            <1>       ; eax < = 32
10524 0000416D 2C08              <1>       sub    al, 8 ; 8->0, 16->8, 24->16, 32->24
10525 0000416F BE[06740000]      <1>       mov    esi, direct_color_fields
10526 00004174 01C6              <1>       add    esi, eax
10527 00004176 56                <1>       push   esi ; ******
10528 00004177 AD                <1>       lodsd  ; RedMaskSize (AL), RedFieldPosition (AH)
10529                            <1>              ; GreenMaskSize (16), GreenFieldPosition (24)
10530 00004178 AB                <1>       stosd
10531 00004179 AD                <1>       lodsd  ; BlueMaskSize (AL), BlueFieldPosition (AH)
10532                            <1>              ; RsvdMaskSize (16), RsvdFieldPosition (24)
10533 0000417A AB                <1>       stosd
10534 0000417B 5E                <1>       pop    esi ; ******
10535                            <1>
10536 0000417C 30C0              <1>       xor    al, al ; 0
10537 0000417E 80F920            <1>       cmp    cl, 32
10538 00004181 7202              <1>       jb     short sml_5
10539 00004183 B002              <1>       mov    al, VBE_DIRECTCOLOR_RESERVED_BITS_AVAILABLE
10540                            <1> sml_5:
10541 00004185 AA                <1>       stosb  ; DirectColorModeInfo
10542                            <1>
10543                            <1>       ; // Mandatory information for VBE 2.0 and above
10544                            <1>
10545 00004186 B8000000E0        <1>       mov    eax, VBE_DISPI_LFB_PHYSICAL_ADDRESS
10546 0000418B AB                <1>       stosd  ; PhysBasePtr
10547 0000418C 29C0              <1>       sub    eax, eax
10548 0000418E AB                <1>       stosd  ; OffScreenMemOffset = 0
10549 0000418F 66AB              <1>       stosw  ; OffScreenMemSize = 0
10550                            <1>
10551                            <1>       ;// Mandatory information for VBE 3.0 and above
10552                            <1>
10553                            <1>       ; ebx = pitch
10554 00004191 6689D8            <1>       mov    ax, bx
10555                            <1>       ;stosw
10556                            <1>
```

```
10557                                  <1>       ;xor   al, al
10558                                  <1>       ;stosb ; BnkNumberOfPages = 0
10559                                  <1>       ;stosb ; LinNumberOfPages = 0
10560                                  <1>
10561 00004194 AB                       <1>       stosd  ; pitch (word), 0 (byte), 0 (byte)
10562                                  <1>
10563 00004195 AD                       <1>       lodsd  ; LinRedMaskSize (AL), LinRedFieldPosition (AH)
10564                                  <1>              ; LinGreenMaskSize (16), LinGreenFieldPosition (24)
10565 00004196 AB                       <1>       stosd
10566 00004197 AD                       <1>       lodsd  ; LinBlueMaskSize (AL), LinBlueFieldPosition (AH)
10567                                  <1>              ; LinRsvdMaskSize (16), LinRsvdFieldPosition (24)
10568 00004198 AB                       <1>       stosd
10569                                  <1>
10570 00004199 29C0                     <1>       sub    eax, eax
10571 0000419B AB                       <1>       stosd  ; MaxPixelClock = 0
10572                                  <1>
10573                                  <1>       ;mov   eax, MODE_INFO_LIST
10574                                  <1>       ; 11/12/2020
10575 0000419C BE[3A120300]             <1>       mov    esi, MODE_INFO_LIST
10576                                  <1>
10577 000041A1 C3                       <1>       retn
10578                                  <1>
10579                                  <1> ; end of set_mode_info_list ; edi = set_mode_info_list + 68
10580                                  <1>
10581                                  <1> pci_get_lfb_addr:
10582                                  <1>       ; 11/12/2020
10583                                  <1>       ; Get linear frame buffer base from PCI
10584                                  <1>       ; (vgabios.c, 02/01/2020, vruppert)
10585                                  <1>       ;
10586                                  <1>       ; Input:
10587                                  <1>       ;      ax = PCI device vendor id
10588                                  <1>       ; Output:
10589                                  <1>       ;      ax  = LFB address (high 16 bit) (zf=0)
10590                                  <1>       ;      eax = 0 -> not found (error) (zf=1)
10591                                  <1>       ;
10592                                  <1>       ; Modified registers: eax
10593                                  <1>
10594 000041A2 53                       <1>       push   ebx
10595 000041A3 51                       <1>       push   ecx
10596 000041A4 52                       <1>       push   edx
10597                                  <1>       ;
10598 000041A5 89C3                     <1>       mov    ebx, eax
10599 000041A7 31C9                     <1>       xor    ecx, ecx
10600 000041A9 28D2                     <1>       sub    dl, dl ; mov dl, 0
10601 000041AB E842000000               <1>       call   pci_read_reg
10602 000041B0 6683F8FF                 <1>       cmp    ax, 0FFFFh
10603 000041B4 7417                     <1>       je     short pci_get_lfb_addr_fail
10604                                  <1> pci_get_lfb_addr_next_dev:
10605 000041B6 28D2                     <1>       sub    dl, dl ; mov dl, 0
10606 000041B8 E835000000               <1>       call   pci_read_reg
10607 000041BD 6639D8                   <1>       cmp    ax, bx ; check vendor
10608 000041C0 740F                     <1>       je     short pci_get_lfb_addr_found
10609 000041C2 6683C108                 <1>       add    cx, 08h
10610 000041C6 6681F90002               <1>       cmp    cx, 200h ; search bus 0 and 1
10611 000041CB 72E9                     <1>       jb     short pci_get_lfb_addr_next_dev
10612                                  <1> pci_get_lfb_addr_fail:
10613 000041CD 31C0                     <1>       xor    eax, eax ; no LFB
10614                                  <1>       ; zf = 1
10615 000041CF EB1D                     <1>       jmp    short pci_get_lfb_addr_return
10616                                  <1> pci_get_lfb_addr_found:
10617 000041D1 B210                     <1>       mov    dl, 10h      ; I/O space 0
10618 000041D3 E81A000000               <1>       call   pci_read_reg
10619 000041D8 66A9F1FF                 <1>       test   ax, 0FFF1h
10620 000041DC 740D                     <1>       jz     short pci_get_lfb_addr_success
10621 000041DE B214                     <1>       mov    dl, 14h ; I/O space 1
10622 000041E0 E80D000000               <1>       call   pci_read_reg
10623 000041E5 66A9F1FF                 <1>       test   ax, 0FFF1h
10624 000041E9 75E2                     <1>       jnz    short pci_get_lfb_addr_fail
10625                                  <1> pci_get_lfb_addr_success:
10626 000041EB C1E810                   <1>       shr    eax, 16 ; LFB address (hw)
10627                                  <1>       ; zf = 0
10628                                  <1> pci_get_lfb_addr_return:
10629 000041EE 5A                       <1>       pop    edx
10630 000041EF 59                       <1>       pop    ecx
10631 000041F0 5B                       <1>       pop    ebx
10632 000041F1 C3                       <1>       retn
10633                                  <1>
10634                                  <1> pci_read_reg:
10635                                  <1>       ; 11/12/2020
10636                                  <1>       ; Read PCI register
10637                                  <1>       ; (vgabios.c, 02/01/2020, vruppert)
10638                                  <1>       ;
10639                                  <1>       ; Input:
10640                                  <1>       ;      cx = device/function
10641                                  <1>       ;      dl = register
10642                                  <1>       ; Output:
10643                                  <1>       ;      eax = value
10644                                  <1>       ;
10645                                  <1>       ; Modified registers: eax, edx
10646                                  <1>
10647 000041F2 B800008000               <1>       mov    eax, 00800000h
10648 000041F7 6689C8                   <1>       mov    ax, cx
10649 000041FA C1E008                   <1>       shl    eax, 8
10650 000041FD 88D0                     <1>       mov    al, dl
10651 000041FF 66BAF80C                 <1>       mov    dx, 0CF8h
10652 00004203 EF                       <1>       out    dx, eax
10653 00004204 80C204                   <1>       add    dl, 4 ; mov dx, 0CFCh
10654 00004207 ED                       <1>       in     eax, dx
10655 00004208 C3                       <1>       retn
10656                                  <1>
10657                                  <1> %endif
10658                                  <1>
10659                                  <1> ; -----------
10660                                  <1>
10661                                  <1> %if 0
```

```
10662                                  <1>
10663                                  <1> mode_info_find_mode:
10664                                  <1>       ; 25/11/2020
10665                                  <1>       ; Input:
10666                                  <1>       ;      bx = VESA VBE2 video mode (+ bochs extensions)
10667                                  <1>       ; Output:
10668                                  <1>       ;      esi = mode info address (for BX input)
10669                                  <1>       ;      esi = 0 -> not found
10670                                  <1>       ;
10671                                  <1>       ; Modified registers: eax, esi
10672                                  <1>
10673                                  <1>       xor    eax, eax
10674                                  <1>       mov    esi, MODE_INFO_LIST
10675                                  <1> mifm_1:
10676                                  <1>       mov    ax, [esi]
10677                                  <1>       cmp    ax, bx
10678                                  <1>       je     short mifm_2
10679                                  <1>       add    esi, MODEINFO.size ; add esi, 68
10680                                  <1>       cmp    esi, VBE_VESA_MODE_END_OF_LIST
10681                                  <1>       jb     short mifm_1
10682                                  <1>       ; not found
10683                                  <1>       sub    esi, esi ; 0
10684                                  <1> mifm_2
10685                                  <1>       retn
10686                                  <1>
10687                                  <1> dispi_get_bpp:
10688                                  <1>       ; 28/11/2020
10689                                  <1>       ; Input:
10690                                  <1>       ;      none
10691                                  <1>       ; Output:
10692                                  <1>       ;      al = Bits per pixel
10693                                  <1>       ;           (8,16,24,32)
10694                                  <1>       ;      ah = Bytes per pixel
10695                                  <1>       ;           (1,2,3,4)
10696                                  <1>       ;
10697                                  <1>       ; Modified registers: none
10698                                  <1>
10699                                  <1>       ;push  edx
10700                                  <1>       ;mov   dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
10701                                  <1>       ;mov   ax, 03h        ; VBE_DISPI_INDEX_BPP
10702                                  <1>       ;out   dx, ax
10703                                  <1>       ;;mov  dx, 01CFh ; VBE_DISPI_IOPORT_DATA
10704                                  <1>       ;;mov  dl, 0CFh
10705                                  <1>       ;inc   dl
10706                                  <1>       ;in    ax, dx
10707                                  <1>       ;mov   ah, al
10708                                  <1>       ;shr   ah, 3 ; / 8
10709                                  <1>       ;;test al, 7 ; 15 bit graphics mode
10710                                  <1>       ;;jz   short get_bpp_noinc
10711                                  <1>       ;;inc  ah
10712                                  <1> ;;get_bpp_noinc:
10713                                  <1>       ;pop   edx
10714                                  <1>       ;retn
10715                                  <1>
10716                                  <1>       ; 28/11/2020
10717                                  <1>       ; Modified registers: edx
10718                                  <1>       ;push  edx
10719                                  <1>       mov    dx, 03h ; VBE_DISPI_INDEX_BPP
10720                                  <1>       call   dispi_get_parms
10721                                  <1>       ;pop   edx
10722                                  <1>       ;retn
10723                                  <1>       mov    ah, al
10724                                  <1>       shr    ah, 3 ; / 8
10725                                  <1>       ;test  al, 7 ; 15 bit graphips mode
10726                                  <1>       ;jz    short get_bpp_noinc
10727                                  <1>       ;inc   ah
10728                                  <1> ;get_bpp_noinc:
10729                                  <1>       ;pop   edx
10730                                  <1>       retn
10731                                  <1>
10732                                  <1> restore_vesa_video_state:
10733                                  <1>       ; 14/01/2021
10734                                  <1>       ; 06/12/2020
10735                                  <1>       ; Input:
10736                                  <1>       ;      [vbe3stbufsize] <= 32 ; <= 32*64 bytes
10737                                  <1>       ; Output:
10738                                  <1>       ;      AX = 004Fh (successed)
10739                                  <1>       ;
10740                                  <1>       ;      eax = 0 -> buffer size problem
10741                                  <1>       ;      eax > 0 and ax <> 004Fh -> failed
10742                                  <1>       ;
10743                                  <1>       ; Modified regs: eax, ebx, ecx, edx, esi, edi
10744                                  <1>
10745                                  <1>       ;movzx ecx, word [vbe3stbufsize]
10746                                  <1>       ;cmp   cx, 32 ; 32 * 64 bytes
10747                                  <1>       ;ja    short r_v_b_s_fail
10748                                  <1>
10749                                  <1>       movzx  ecx, byte [vbe3stbufsize]; <=32
10750                                  <1>       shl    cx, 4 ; dword count for movsd
10751                                  <1>       mov    edi, VBE3SAVERESTOREBLOCK ; destination
10752                                  <1>                                 ; (vbe3 pmi buff)
10753                                  <1>       mov    esi, VBE3VIDEOSTATE ; source (kernel buff)
10754                                  <1>       rep    movsd
10755                                  <1>
10756                                  <1>       mov    ax, 4F04h
10757                                  <1>       mov    dl, 02h ; restore
10758                                  <1>       ;mov   cx, 0Fh
10759                                  <1>       mov    cl, 0Fh
10760                                  <1>       xor    ebx, ebx ; points to VBE3SAVERESTOREBLOCK
10761                                  <1>       jmp    short int10h_32bit_pmi
10762                                  <1>
10763                                  <1> ;s_v_b_s_fail:
10764                                  <1> ;r_v_b_s_fail:
10765                                  <1> ;     xor    eax, eax
10766                                  <1> ;     retn
```

```
10767                                  <1>
10768                                  <1> save_vesa_video_state:
10769                                  <1>       ; 14/01/2021
10770                                  <1>       ; 06/12/2020
10771                                  <1>       ; Input:
10772                                  <1>       ;      [vbe3stbufsize] <= 32 ; <= 32*64 bytes
10773                                  <1>       ; Output:
10774                                  <1>       ;      AX = 004Fh (successed)
10775                                  <1>       ;
10776                                  <1>       ;      eax = 0 -> buffer size problem
10777                                  <1>       ;      eax > 0 and ax <> 004Fh -> failed
10778                                  <1>       ;
10779                                  <1>       ; Modified regs: eax, ebx, ecx, edx, esi, edi
10780                                  <1>
10781                                  <1>       ;cmp   word [vbe33stbufsize], 32
10782                                  <1>       ;                    ; 32 * 64 bytes
10783                                  <1>       ;ja    short s_v_b_s_fail
10784                                  <1>
10785                                  <1>       mov    ax, 4F04h
10786                                  <1>       mov    dl, 01h ; save
10787                                  <1>       ;mov   cx, 0Fh
10788                                  <1>       mov    cl, 0Fh
10789                                  <1>       xor    ebx, ebx ; points to VBE3SAVERESTOREBLOCK
10790                                  <1>
10791                                  <1>       call   int10h_32bit_pmi
10792                                  <1>
10793                                  <1>       movzx  ecx, byte [vbe3stbufsize]; <=32
10794                                  <1>       shl    cx, 4 ; dword count for movsd
10795                                  <1>       mov    esi, VBE3SAVERESTOREBLOCK ; destination
10796                                  <1>                                  ; (vbe3 pmi buff)
10797                                  <1>       mov    edi, VBE3VIDEOSTATE ; source (kernel buff)
10798                                  <1>       rep    movsd
10799                                  <1>       retn
10800                                  <1>
10801                                  <1> dispi_set_bank_farcall:
10802                                  <1>       ; 11/12/2020
10803                                  <1>       ; (This may be 'sysvideo' function, later)
10804                                  <1>       ;
10805                                  <1>       ; Input:
10806                                  <1>       ;      bx = 0000h, set bank number
10807                                  <1>       ;         = 0100h, get bank number
10808                                  <1>       ;      dx = bank number (if bx = 0)
10809                                  <1>       ; Output:
10810                                  <1>       ;      dx = bank number
10811                                  <1>
10812                                  <1>       cmp    bx, 0100h
10813                                  <1>       je     short dispi_set_bank_farcall_get
10814                                  <1>       or     bx, bx
10815                                  <1>       jnz    dispi_set_bank_farcall_error
10816                                  <1>       mov    ax, dx
10817                                  <1>       push   dx
10818                                  <1>       push   ax
10819                                  <1>       mov    ax, VBE_DISPI_INDEX_BANK
10820                                  <1>       mov    dx, VBE_DISPI_IOPORT_INDEX
10821                                  <1>       out    dx, ax
10822                                  <1>       pop    ax
10823                                  <1>       mov    dx, VBE_DISPI_IOPORT_DATA
10824                                  <1>       out    dx, ax
10825                                  <1>       in     ax, dx
10826                                  <1>       pop    dx
10827                                  <1>       cmp    dx, ax
10828                                  <1>       jne    short dispi_set_bank_farcall_error
10829                                  <1>       mov    ax, 004Fh
10830                                  <1>       retn   ; retf for real mode far call
10831                                  <1> dispi_set_bank_farcall_get:
10832                                  <1>       mov    ax, VBE_DISPI_INDEX_BANK
10833                                  <1>       mov    dx, VBE_DISPI_IOPORT_INDEX
10834                                  <1>       out    dx, ax
10835                                  <1>       mov    dx, VBE_DISPI_IOPORT_DATA
10836                                  <1>       in     ax, dx
10837                                  <1>       mov    dx, ax
10838                                  <1>       retn   ; retf for real mode far call
10839                                  <1> dispi_set_bank_farcall_error:
10840                                  <1>       mov    ax, 014Fh
10841                                  <1>       retn   ; retf for real mode far call
10842                                  <1>
10843                                  <1> %endif
10844                                  <1>
10845                                  <1> ; % include 'vidata.s' ; VIDEO DATA
10846                                  <1>
10847                                  <1> ; /// End Of VIDEO FUNCTIONS ///
 2655
 2656                                      setup_rtc_int:
 2657                                      ; source: http://wiki.osdev.org/RTC
 2658 00004209 FA                         cli          ; disable interrupts
 2659                                     ; default int frequency is 1024 Hz (Lower 4 bits of register A is 0110b or 6)
 2660                                     ; in order to change this ...
 2661                                     ; frequency  = 32768 >> (rate-1) --> 32768 >> 5 = 1024
 2662                                     ; (rate must be above 2 and not over 15)
 2663                                     ; new rate = 15 --> 32768 >> (15-1) = 2 Hz
 2664 0000420A B08A                      mov    al, 8Ah
 2665 0000420C E670                      out    70h, al ; set index to register A, disable NMI
 2666 0000420E 90                        nop
 2667 0000420F E471                      in     al, 71h ; get initial value of register A
 2668 00004211 88C4                      mov    ah, al
 2669 00004213 80E4F0                    and    ah, 0F0h
 2670 00004216 B08A                      mov    al, 8Ah
 2671 00004218 E670                      out    70h, al ; reset index to register A
 2672 0000421A 88E0                      mov    al, ah
 2673 0000421C 0C0F                      or     al, 0Fh      ; new rate (0Fh -> 15)
 2674 0000421E E671                      out    71h, al ; write only our rate to A. Note, rate is the bottom 4 bits.
 2675                                     ; enable RTC interrupt
 2676 00004220 B08B                      mov    al, 8Bh ;
 2677 00004222 E670                      out    70h, al ; select register B and disable NMI
 2678 00004224 90                        nop
```

```
2679 00004225 E471                      in      al, 71h ; read the current value of register B
2680 00004227 88C4                      mov     ah, al  ;
2681 00004229 B08B                      mov     al, 8Bh ;
2682 0000422B E670                      out     70h, al ; set the index again (a read will reset the index to register B)
2683 0000422D 88E0                      mov     al, ah  ;
2684 0000422F 0C40                      or      al, 40h ;
2685 00004231 E671                      out     71h, al ; write the previous value ORed with 0x40. This turns on bit 6 of register B
2686 00004233 FB                        sti
2687 00004234 C3                        retn
2688
2689                                ; Write memory information
2690                                ; 29/01/2016
2691                                ; 06/11/2014
2692                                ; 14/08/2015
2693                                memory_info:
2694 00004235 A1[94810100]                  mov     eax, [memory_size] ; in pages
2695 0000423A 50                            push    eax
2696 0000423B C1E00C                        shl     eax, 12                 ; in bytes
2697 0000423E BB0A000000                    mov     ebx, 10
2698 00004243 89D9                          mov     ecx, ebx        ; 10
2699 00004245 BE[19440100]                  mov     esi, mem_total_b_str
2700 0000424A E8D7000000                    call    bintdstr
2701 0000424F 58                            pop     eax
2702 00004250 B107                          mov     cl, 7
2703 00004252 BE[3D440100]                  mov     esi, mem_total_p_str
2704 00004257 E8CA000000                    call    bintdstr
2705                                        ; 14/08/2015
2706 0000425C E8E2000000                    call    calc_free_mem
2707                                        ; edx = calculated free pages
2708                                        ; ecx = 0
2709 00004261 A1[98810100]                  mov     eax, [free_pages]
2710 00004266 39D0                          cmp     eax, edx ; calculated free mem value
2711                                                 ; and initial free mem value are same or not?
2712 00004268 751D                          jne     short pmim ; print mem info with '?' if not
2713 0000426A 52                            push    edx ; free memory in pages
2714                                        ;mov     eax, edx
2715 0000426B C1E00C                        shl     eax, 12 ; convert page count
2716                                                 ; to byte count
2717 0000426E B10A                          mov     cl, 10
2718 00004270 BE[5D440100]                  mov     esi, free_mem_b_str
2719 00004275 E8AC000000                    call    bintdstr
2720 0000427A 58                            pop     eax
2721 0000427B B107                          mov     cl, 7
2722 0000427D BE[81440100]                  mov     esi, free_mem_p_str
2723 00004282 E89F000000                    call    bintdstr
2724                                pmim:
2725 00004287 BE[07440100]                  mov     esi, msg_memory_info
2726                                        ;
2727 0000428C B407                          mov     ah, 07h ; Black background,
2728                                                 ; light gray forecolor
2729                                print_kmsg: ; 29/01/2016
2730 0000428E 8825[BF810100]                mov     [ccolor], ah
2731                                pkmsg_loop:
2732 00004294 AC                            lodsb
2733 00004295 08C0                          or      al, al
2734 00004297 7410                          jz      short pkmsg_ok
2735 00004299 56                            push    esi
2736                                        ; 13/05/2016
2737 0000429A 0FB61D[BF810100]              movzx   ebx, byte [ccolor]
2738                                                 ; Video page 0 (bh=0)
2739 000042A1 E857E0FFFF                    call    _write_tty
2740 000042A6 5E                            pop     esi
2741 000042A7 EBEB                          jmp     short pkmsg_loop
2742                                pkmsg_ok:
2743 000042A9 C3                            retn
2744
2745                                ; 19/12/2020
2746                                ; temporary
2747                                ; Write default liner frame buffer address
2748                                ;
2749                                default_lfb_info:
2750 000042AA 66A1[E30E0000]                mov     ax, [def_LFB_addr] ; high word
2751 000042B0 E829000000                    call    wordtohex
2752 000042B5 A3[D9440100]                  mov     dword [lfb_addr_str], eax
2753 000042BA BE[C2440100]                  mov     esi, msg_lfb_addr
2754 000042BF B40F                          mov     ah, 0Fh ; Black background,
2755                                                 ; white forecolor
2756 000042C1 EBCB                          jmp     short print_kmsg
2757
2758                                ; Convert binary number to hexadecimal string
2759                                ; 10/05/2015
2760                                ; dsectpm.s (28/02/2015)
2761                                ; Retro UNIX 386 v1 - Kernel v0.2.0.6
2762                                ; 01/12/2014
2763                                ; 25/11/2014
2764                                ;
2765                                bytetohex:
2766                                      ; INPUT ->
2767                                      ;      AL = byte (binary number)
2768                                      ; OUTPUT ->
2769                                      ;      AX = hexadecimal string
2770                                      ;
2771 000042C3 53                            push    ebx
2772 000042C4 31DB                          xor     ebx, ebx
2773 000042C6 88C3                          mov     bl, al
2774 000042C8 C0EB04                        shr     bl, 4
2775 000042CB 8A9B[15430000]                mov     bl, [ebx+hexchrs]
2776 000042D1 86D8                          xchg    bl, al
2777 000042D3 80E30F                        and     bl, 0Fh
2778 000042D6 8AA3[15430000]                mov     ah, [ebx+hexchrs]
2779 000042DC 5B                            pop     ebx
2780 000042DD C3                            retn
2781
2782                                wordtohex:
2783                                      ; INPUT ->
```

```
                                        ;       AX = word (binary number)
                                        ; OUTPUT ->
                                        ;       EAX = hexadecimal string
                                        ;
000042DE 53                             push    ebx
000042DF 31DB                           xor     ebx, ebx
000042E1 86E0                           xchg    ah, al
000042E3 6650                           push    ax
000042E5 88E3                           mov     bl, ah
000042E7 C0EB04                         shr     bl, 4
000042EA 8A83[15430000]                 mov     al, [ebx+hexchrs]
000042F0 88E3                           mov     bl, ah
000042F2 80E30F                         and     bl, 0Fh
000042F5 8AA3[15430000]                 mov     ah, [ebx+hexchrs]
000042FB C1E010                         shl     eax, 16
000042FE 6658                           pop     ax
00004300 5B                             pop     ebx
00004301 EBC0                           jmp     short bytetohex
                                        ;mov    bl, al
                                        ;shr    bl, 4
                                        ;mov    bl, [ebx+hexchrs]
                                        ;xchg   bl, al
                                        ;and    bl, 0Fh
                                        ;mov    ah, [ebx+hexchrs]
                                        ;pop    ebx
                                        ;retn

                                dwordtohex:
                                        ; INPUT ->
                                        ;       EAX = dword (binary number)
                                        ; OUTPUT ->
                                        ;       EDX:EAX = hexadecimal string
                                        ;
00004303 50                             push    eax
00004304 C1E810                         shr     eax, 16
00004307 E8D2FFFFFF                     call    wordtohex
0000430C 89C2                           mov     edx, eax
0000430E 58                             pop     eax
0000430F E8CAFFFFFF                     call    wordtohex
00004314 C3                             retn

                                ; 10/05/2015
                                hex_digits:
                                hexchrs:
00004315 303132333435363738-            db '0123456789ABCDEF'
0000431E 39414243444546
                                ; 19/01/2021 - VESA EDID ready flag (4Fh)
00004325 00                     edid: db 0

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

                                bintdstr:
                                        ; EAX = binary number
                                        ; ESI = decimal/numeric string address
                                        ; EBX = divisor (10)
                                        ; ECX = string length (<=10)
00004326 01CE                           add     esi, ecx
                                btdstr0:
00004328 4E                             dec     esi
00004329 31D2                           xor     edx, edx
0000432B F7F3                           div     ebx
0000432D 80C230                         add     dl, 30h
00004330 8816                           mov     [esi], dl
00004332 FEC9                           dec     cl
00004334 740C                           jz      short btdstr2 ; 08/09/2016
00004336 09C0                           or      eax, eax
00004338 75EE                           jnz     short btdstr0
                                btdstr1:
0000433A 4E                             dec     esi
0000433B C60620                           mov     byte [esi], 20h ; blank space
0000433E FEC9                           dec     cl
00004340 75F8                           jnz     short btdstr1
                                btdstr2:
00004342 C3                             retn

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

                                calc_free_mem:
00004343 31D2                           xor     edx, edx
                                        ;xor    ecx, ecx
00004345 668B0D[A8810100]               mov     cx, [mat_size] ; in pages
0000434C C1E10A                         shl     ecx, 10       ; 1024 dwords per page
0000434F BE00001000                     mov     esi, MEM_ALLOC_TBL
                                cfm0:
00004354 AD                             lodsd
00004355 51                             push    ecx
00004356 B920000000                     mov     ecx, 32
                                cfm1:
0000435B D1E8                           shr     eax, 1
0000435D 7301                           jnc     short cfm2
0000435F 42                             inc     edx
                                cfm2:
00004360 E2F9                           loop    cfm1
00004362 59                             pop     ecx
00004363 E2EF                           loop    cfm0
00004365 C3                             retn

                                %include 'diskio.s'  ; 07/03/2015
                              <1> ; ****************************************************************************
```

```
                        <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.2 - diskio.s
                        <1> ; ----------------------------------------------------------------------------
                        <1> ; Last Update: 30/08/2020
                        <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> ; diskio.inc (22/08/2015)
                        <1> ;
                        <1> ; Derived from 'IBM PC-XT-286' BIOS source code (1986)
                        <1> ; ****************************************************************************
                        <1>
                        <1> ; Retro UNIX 386 v1 Kernel - DISKIO.INC
                        <1> ; Last Modification: 22/08/2015
                        <1> ;     (Initialized Disk Parameters Data is in 'DISKDATA.INC')
                        <1> ;     (Uninitialized Disk Parameters Data is in 'DISKBSS.INC')
                        <1>
                        <1> ; DISK I/O SYSTEM - Erdogan Tan (Retro UNIX 386 v1 project)
                        <1>
                        <1> ; ///////// DISK I/O SYSTEM ///////////////
                        <1>
                        <1> ; 06/02/2015
                        <1> diskette_io:
00004366 F8             <1>       clc ; 20/07/2020
00004367 9C             <1>       pushfd
00004368 0E             <1>       push   cs
00004369 E809000000     <1>       call   DISKETTE_IO_1
0000436E C3             <1>       retn
                        <1>
                        <1> ;;;;;; DISKETTE I/O ;;;;;;;;;;;;;;;;;;; 06/02/2015 ;;;
                        <1> ;////////////////////////////////////////////////////
                        <1>
                        <1> ; DISKETTE I/O - Erdogan Tan (Retro UNIX 386 v1 project)
                        <1> ; 20/02/2015
                        <1> ; 06/02/2015 (unix386.s)
                        <1> ; 16/12/2014 - 02/01/2015 (dsectrm2.s)
                        <1> ;
                        <1> ; Code (DELAY) modifications - AWARD BIOS 1999 (ADISK.EQU, COMMON.MAC)
                        <1> ;
                        <1> ; ADISK.EQU
                        <1>
                        <1> ;----- Wait control constants
                        <1>
                        <1> ;amount of time to wait while RESET is active.
                        <1>
                        <1> WAITCPU_RESET_ON   EQU    21          ;Reset on must last at least 14us
                        <1>                                       ;at 250 KBS xfer rate.
                        <1>                                       ;see INTEL MCS, 1985, pg. 5-456
                        <1>
                        <1> WAITCPU_FOR_STATUS EQU    100         ;allow 30 microseconds for
                        <1>                                       ;status register to become valid
                        <1>                                       ;before re-reading.
                        <1>
                        <1> ;After sending a byte to NEC, status register may remain
                        <1> ;incorrectly set for 24 us.
                        <1>
                        <1> WAITCPU_RQM_LOW          EQU    24          ;number of loops to check for
                        <1>                                       ;RQM low.
                        <1>
                        <1> ; COMMON.MAC
                        <1> ;
                        <1> ;     Timing macros
                        <1> ;
                        <1>
                        <1> %macro             SIODELAY 0               ; SHORT IODELAY
                        <1>             jmp short $+2
                        <1> %endmacro
                        <1>
                        <1> %macro             IODELAY  0               ; NORMAL IODELAY
                        <1>             jmp short $+2
                        <1>             jmp short $+2
                        <1> %endmacro
                        <1>
                        <1> %macro             NEWIODELAY 0
                        <1>             out    0ebh,al
                        <1> %endmacro
                        <1>
                        <1> ; (According to) AWARD BIOS 1999 - ATORGS.ASM (dw -> equ, db -> equ)
                        <1> ;;; WAIT_FOR_MEM
                        <1> ;WAIT_FDU_INT_LO   equ    017798       ; 2.5 secs in 30 micro units.
                        <1> ;WAIT_FDU_INT_HI   equ    1
                        <1> WAIT_FDU_INT_LH          equ    83334        ; 27/02/2015 (2.5 seconds waiting)
                        <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)
```

```
107                                     <1>
108                                     <1> ;----------------------------------------
109                                     <1> ;     EQUATES USED BY POST AND BIOS    :
110                                     <1> ;----------------------------------------
111                                     <1>
112                                     <1> ;--------- 8042 KEYBOARD INTERFACE AND DIAGNOSTIC CONTROL REGISTERS ------------
113                                     <1> ;PORT_A            EQU    060H         ; 8042 KEYBOARD SCAN CODE/CONTROL PORT
114                                     <1> ;PORT_B            EQU    061H         ; PORT B READ/WRITE DIAGNOSTIC REGISTER
115                                     <1> ;REFRESH_BIT EQU    00010000B    ; REFRESH TEST BIT
116                                     <1>
117                                     <1> ;----------------------------------------
118                                     <1> ;     CMOS EQUATES FOR THIS SYSTEM     :
119                                     <1> ;--------------------------------------------------------------------------------
120                                     <1> ;CMOS_PORT   EQU    070H         ; I/O ADDRESS OF CMOS ADDRESS PORT
121                                     <1> ;CMOS_DATA   EQU    071H         ; I/O ADDRESS OF CMOS DATA PORT
122                                     <1> ;NMI         EQU    10000000B    ; DISABLE NMI INTERRUPTS MASK -
123                                     <1>                                 ;  HIGH BIT OF CMOS LOCATION ADDRESS
124                                     <1>
125                                     <1> ;---------- CMOS TABLE LOCATION ADDRESS'S ## ---------------------------------
126                                     <1> CMOS_DISKETTE      EQU    010H         ; DISKETTE DRIVE TYPE BYTE      ;
127                                     <1> ;            EQU    011H         ; - RESERVED                    ;C
128                                     <1> CMOS_DISK    EQU    012H         ; FIXED DISK TYPE BYTE                  ;H
129                                     <1> ;            EQU    013H         ; - RESERVED                    ;E
130                                     <1> CMOS_EQUIP   EQU    014H         ; EQUIPMENT WORD LOW BYTE       ;C
131                                     <1>
132                                     <1> ;---------- DISKETTE EQUATES -----------------------------------------------
133                                     <1> INT_FLAG     EQU    10000000B    ; INTERRUPT OCCURRENCE FLAG
134                                     <1> DSK_CHG      EQU    10000000B    ; DISKETTE CHANGE FLAG MASK BIT
135                                     <1> DETERMINED   EQU    00010000B    ; SET STATE DETERMINED IN STATE BITS
136                                     <1> HOME         EQU    00010000B    ; TRACK 0 MASK
137                                     <1> SENSE_DRV_ST EQU    00000100B    ; SENSE DRIVE STATUS COMMAND
138                                     <1> TRK_SLAP     EQU    030H         ; CRASH STOP (48 TPI DRIVES)
139                                     <1> QUIET_SEEK   EQU    00AH         ; SEEK TO TRACK 10
140                                     <1> ;MAX_DRV     EQU    2            ; MAX NUMBER OF DRIVES
141                                     <1> HD12_SETTLE  EQU    15           ; 1.2 M HEAD SETTLE TIME
142                                     <1> HD320_SETTLE EQU    20           ; 320 K HEAD SETTLE TIME
143                                     <1> MOTOR_WAIT   EQU    37           ; 2 SECONDS OF COUNTS FOR MOTOR TURN OFF
144                                     <1>
145                                     <1> ;---------- DISKETTE ERRORS ------------------------------------------------
146                                     <1> ;TIME_OUT    EQU    080H         ; ATTACHMENT FAILED TO RESPOND
147                                     <1> ;BAD_SEEK    EQU    040H         ; SEEK OPERATION FAILED
148                                     <1> BAD_NEC      EQU    020H         ; DISKETTE CONTROLLER HAS FAILED
149                                     <1> BAD_CRC      EQU    010H         ; BAD CRC ON DISKETTE READ
150                                     <1> MED_NOT_FND  EQU    00CH         ; MEDIA TYPE NOT FOUND
151                                     <1> DMA_BOUNDARY EQU    009H         ; ATTEMPT TO DMA ACROSS 64K BOUNDARY
152                                     <1> BAD_DMA      EQU    008H         ; DMA OVERRUN ON OPERATION
153                                     <1> MEDIA_CHANGE EQU    006H         ; MEDIA REMOVED ON DUAL ATTACH CARD
154                                     <1> RECORD_NOT_FND     EQU    004H         ; REQUESTED SECTOR NOT FOUND
155                                     <1> WRITE_PROTECT      EQU    003H         ; WRITE ATTEMPTED ON WRITE PROTECT DISK
156                                     <1> BAD_ADDR_MARK      EQU    002H         ; ADDRESS MARK NOT FOUND
157                                     <1> BAD_CMD      EQU    001H         ; BAD COMMAND PASSED TO DISKETTE I/O
158                                     <1>
159                                     <1> ;---------- DISK CHANGE LINE EQUATES ----------------------------------------
160                                     <1> NOCHGLN      EQU    001H         ; NO DISK CHANGE LINE AVAILABLE
161                                     <1> CHGLN        EQU    002H         ; DISK CHANGE LINE AVAILABLE
162                                     <1>
163                                     <1> ;---------- MEDIA/DRIVE STATE INDICATORS ------------------------------------
164                                     <1> TRK_CAPA     EQU    00000001B    ; 80 TRACK CAPABILITY
165                                     <1> FMT_CAPA     EQU    00000010B    ; MULTIPLE FORMAT CAPABILITY (1.2M)
166                                     <1> DRV_DET      EQU    00000100B    ; DRIVE DETERMINED
167                                     <1> MED_DET      EQU    00010000B    ; MEDIA DETERMINED BIT
168                                     <1> DBL_STEP     EQU    00100000B    ; DOUBLE STEP BIT
169                                     <1> RATE_MSK     EQU    11000000B    ; MASK FOR CLEARING ALL BUT RATE
170                                     <1> RATE_500     EQU    00000000B    ; 500 KBS DATA RATE
171                                     <1> RATE_300     EQU    01000000B    ; 300 KBS DATA RATE
172                                     <1> RATE_250     EQU    10000000B    ; 250 KBS DATA RATE
173                                     <1> STRT_MSK     EQU    00001100B    ; OPERATION START RATE MASK
174                                     <1> SEND_MSK     EQU    11000000B    ; MASK FOR SEND RATE BITS
175                                     <1>
176                                     <1> ;---------- MEDIA/DRIVE STATE INDICATORS COMPATIBILITY ------------------------
177                                     <1> M3D3U        EQU    00000000B    ; 360 MEDIA/DRIVE NOT ESTABLISHED
178                                     <1> M3D1U        EQU    00000001B    ; 360 MEDIA,1.2DRIVE NOT ESTABLISHED
179                                     <1> M1D1U        EQU    00000010B    ; 1.2 MEDIA/DRIVE NOT ESTABLISHED
180                                     <1> MED_UNK      EQU    00000111B    ; NONE OF THE ABOVE
181                                     <1>
182                                     <1> ;---------- INTERRUPT EQUATES -----------------------------------------------
183                                     <1> ;EOI         EQU    020H         ; END OF INTERRUPT COMMAND TO 8259
184                                     <1> ;INTA00            EQU    020H         ; 8259 PORT
185                                     <1> INTA01             EQU    021H         ; 8259 PORT
186                                     <1> INTB00             EQU    0A0H         ; 2ND 8259
187                                     <1> INTB01             EQU    0A1H         ;
188                                     <1>
189                                     <1> ;--------------------------------------------------------------------------------
190                                     <1> DMA08        EQU    008H         ; DMA STATUS REGISTER PORT ADDRESS
191                                     <1> DMA          EQU    000H         ; DMA CH.0 ADDRESS REGISTER PORT ADDRESS
192                                     <1> DMA18        EQU    0D0H         ; 2ND DMA STATUS PORT ADDRESS
193                                     <1> DMA1         EQU    0C0H         ; 2ND DMA CH.0 ADDRESS REGISTER ADDRESS
194                                     <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
```

```
                        <1> ;     retn
                        <1>
                        <1> ; DSKETTE ----- 04/21/86 DISKETTE BIOS
                        <1> ; (IBM PC XT Model 286 System BIOS Source Code, 04-21-86)
                        <1> ;
                        <1>
                        <1> ;-- INT13H ------------------------------------------------------------------------
                        <1> ; DISKETTE I/O
                        <1> ;     THIS INTERFACE PROVIDES ACCESS TO THE 5 1/4 INCH 360 KB,
                        <1> ;     1.2 MB, 720 KB AND 1.44 MB DISKETTE DRIVES.
                        <1> ; INPUT
                        <1> ;     (AH) =  00H RESET DISKETTE SYSTEM
                        <1> ;           HARD RESET TO NEC, PREPARE COMMAND, RECALIBRATE REQUIRED
                        <1> ;           ON ALL DRIVES
                        <1> ;-------------------------------------------------------------------------------
                        <1> ;     (AH)= 01H  READ THE STATUS OF THE SYSTEM INTO (AH)
                        <1> ;           @DISKETTE_STATUS FROM LAST OPERATION IS USED
                        <1> ;-------------------------------------------------------------------------------
                        <1> ;     REGISTERS FOR READ/WRITE/VERIFY/FORMAT
                        <1> ;     (DL) - DRIVE NUMBER (0-1 ALLOWED, VALUE CHECKED)
                        <1> ;     (DH) - HEAD NUMBER (0-1 ALLOWED, NOT VALUE CHECKED)
                        <1> ;     (CH) - TRACK NUMBER (NOT VALUE CHECKED)
                        <1> ;           MEDIA  DRIVE  TRACK NUMBER
                        <1> ;           320/360      320/360          0-39
                        <1> ;           320/360      1.2M       0-39
                        <1> ;           1.2M   1.2M       0-79
                        <1> ;           720K   720K       0-79
                        <1> ;           1.44M  1.44M      0-79
                        <1> ;     (CL) -       SECTOR NUMBER (NOT VALUE CHECKED, NOT USED FOR FORMAT)
                        <1> ;           MEDIA  DRIVE  SECTOR NUMBER
                        <1> ;           320/360      320/360           1-8/9
                        <1> ;           320/360      1.2M        1-8/9
                        <1> ;           1.2M   1.2M       1-15
                        <1> ;           720K   720K       1-9
                        <1> ;           1.44M  1.44M      1-18
                        <1> ;     (AL)   NUMBER OF SECTORS (NOT VALUE CHECKED)
                        <1> ;           MEDIA  DRIVE  MAX NUMBER OF SECTORS
                        <1> ;           320/360      320/360              8/9
                        <1> ;           320/360      1.2M          8/9
                        <1> ;           1.2M   1.2M         15
                        <1> ;           720K   720K         9
                        <1> ;           1.44M  1.44M        18
                        <1> ;
                        <1> ;     (ES:BX) - ADDRESS OF BUFFER (NOT REQUIRED FOR VERIFY)
                        <1> ;
                        <1> ;-------------------------------------------------------------------------------
                        <1> ;     (AH)= 02H  READ THE DESIRED SECTORS INTO MEMORY
                        <1> ;-------------------------------------------------------------------------------
                        <1> ;     (AH)= 03H  WRITE THE DESIRED SECTORS FROM MEMORY
                        <1> ;-------------------------------------------------------------------------------
                        <1> ;     (AH)= 04H  VERIFY THE DESIRED SECTORS
                        <1> ;-------------------------------------------------------------------------------
                        <1> ;     (AH)= 05H  FORMAT THE DESIRED TRACK
                        <1> ;           (ES,BX) MUST POINT TO THE COLLECTION OF DESIRED ADDRESS FIELDS
                        <1> ;           FOR THE      TRACK. EACH FIELD IS COMPOSED OF 4 BYTES, (C,H,R,N),
                        <1> ;           WHERE C = TRACK NUMBER, H=HEAD NUMBER, R = SECTOR NUMBER,
                        <1> ;           N= NUMBER OF BYTES PER SECTOR (00=128,01=256,02=512,03=1024),
                        <1> ;           THERE MUST BE ONE ENTRY FOR EVERY SECTOR ON THE TRACK.
                        <1> ;           THIS INFORMATION IS USED TO FIND THE REQUESTED SECTOR DURING
                        <1> ;           READ/WRITE ACCESS.
                        <1> ;           PRIOR TO FORMATTING A DISKETTE, IF THERE EXISTS MORE THAN
                        <1> ;           ONE SUPPORTED MEDIA FORMAT TYPE WITHIN THE DRIVE IN QUESTION,
                        <1> ;           THEN "SET DASD TYPE" (INT 13H, AH = 17H) OR 'SET MEDIA TYPE'
                        <1> ;           (INT 13H, AH =  18H) MUST BE CALLED TO SET THE DISKETTE TYPE
                        <1> ;           THAT IS TO BE FORMATTED. IF "SET DASD TYPE" OR "SET MEDIA TYPE"
                        <1> ;           IS NOT CALLED, THE FORMAT ROUTINE WILL ASSUME THE
                        <1> ;           MEDIA FORMAT TO BE THE MAXIMUM CAPACITY OF THE DRIVE.
                        <1> ;
                        <1> ;           THESE PARAMETERS OF DISK BASE MUST BE CHANGED IN ORDER TO
                        <1> ;           FORMAT THE FOLLOWING MEDIAS:
                        <1> ;           ------------------------------------------
                        <1> ;           : MEDIA  :     DRIVE      : PARM 1 : PARM 2 :
                        <1> ;           ------------------------------------------
                        <1> ;           : 320K  : 320K/360K/1.2M :  50H   :   8    :
                        <1> ;           : 360K  : 320K/360K/1.2M :  50H   :   9    :
                        <1> ;           : 1.2M  : 1.2M           :  54H   :  15    :
                        <1> ;           : 720K  : 720K/1.44M     :  50H   :   9    :
                        <1> ;           : 1.44M       : 1.44M          :  6CH   :  18    :
                        <1> ;           ------------------------------------------
                        <1> ;           NOTES: - PARM 1 = GAP LENGTH FOR FORMAT
                        <1> ;                  - PARM 2 = EOT (LAST SECTOR ON TRACK)
                        <1> ;                  - DISK BASE IS POINTED BY DISK POINTER LOCATED
                        <1> ;                   AT ABSOLUTE ADDRESS 0:78.
                        <1> ;                  - WHEN FORMAT OPERATIONS ARE COMPLETE, THE PARAMETERS
                        <1> ;                   SHOULD BE RESTORED TO THEIR RESPECTIVE INITIAL VALUES.
                        <1> ;-------------------------------------------------------------------------------
                        <1> ;    (AH) = 08H READ DRIVE PARAMETERS
                        <1> ;    REGISTERS
                        <1> ;      INPUT
                        <1> ;        (DL) - DRIVE NUMBER (0-1 ALLOWED, VALUE CHECKED)
                        <1> ;         ** 27/05/2016 - TRDOS 386 (TRDOS v2.0) **
                        <1> ;           ** EBX = Buffer address for floppy disk parameters table **
                        <1> ;      OUTPUT
                        <1> ;        (ES:DI) POINTS TO DRIVE PARAMETER TABLE
                        <1> ;        *** TRDOS 386 note: floppy disk parameter table (16 bytes)
                        <1> ;        will be returned to user in EBX, buffer address *** 27/05/2016 ***
                        <1> ;
                        <1> ;        (CH) - LOW ORDER 8 OF 10 BITS MAXIMUM NUMBER OF TRACKS
                        <1> ;        (CL) - BITS 7 & 6 - HIGH ORDER TWO BITS OF MAXIMUM TRACKS
                        <1> ;               BITS 5 THRU 0 - MAXIMUM SECTORS PER TRACK
                        <1> ;        (DH) - MAXIMUM HEAD NUMBER
                        <1> ;        (DL) - NUMBER OF DISKETTE DRIVES INSTALLED
                        <1> ;        (BH) - 0
                        <1> ;        (BL) - BITS 7 THRU 4 - 0
                        <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> ;     |       |       |       |       |               |
405                                     <1> ;     |       |       |       |    RESERVED           |
406                                     <1> ;     |       |       |       |                 PRESENT STATE
407                                     <1> ;     |       |       |       |       000: 360K IN 360K DRIVE UNESTABLISHED
408                                     <1> ;     |       |       |       |       001: 360K IN 1.2M DRIVE UNESTABLISHED
409                                     <1> ;     |       |       |       |       010: 1.2M IN 1.2M DRIVE UNESTABLISHED
410                                     <1> ;     |       |       |       |       011: 360K IN 360K DRIVE ESTABLISHED
411                                     <1> ;     |       |       |       |       100: 360K IN 1.2M DRIVE ESTABLISHED
412                                     <1> ;     |       |       |       |       101: 1.2M IN 1.2M DRIVE ESTABLISHED
413                                     <1> ;     |       |       |       |       110: RESERVED
414                                     <1> ;     |       |       |       |       111: NONE OF THE ABOVE
415                                     <1> ;     |       |       |       |
416                                     <1> ;     |       |       |       ------>       MEDIA/DRIVE ESTABLISHED
417                                     <1> ;     |       |       |
418                                     <1> ;     |       |       --------------->     DOUBLE STEPPING REQUIRED (360K IN 1.2M
419                                     <1> ;     |       |                       DRIVE)
420                                     <1> ;     |       |
421                                     <1> ;     ------------------------------->  DATA TRANSFER RATE FOR THIS DRIVE:
```

```
                                <1> ;
                                <1> ;                                   00: 500 KBS
                                <1> ;                                   01: 300 KBS
                                <1> ;                                   10: 250 KBS
                                <1> ;                                   11: RESERVED
                                <1> ;
                                <1> ;--------------------------------------------------------------------------
                                <1> ; STATE OPERATION STARTED - ABSOLUTE ADDRESS 40:92 (DRIVE A) & 93 (DRIVE B)
                                <1> ;--------------------------------------------------------------------------
                                <1> ; PRESENT CYLINDER NUMBER - ABSOLUTE ADDRESS 40:94 (DRIVE A) & 95 (DRIVE B)
                                <1> ;--------------------------------------------------------------------------
                                <1>
                                <1> struc MD
00000000 <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:
0000436F 9C                     <1>       pushfd
00004370 0E                     <1>       push   cs
00004371 E801000000             <1>       call   DISKETTE_IO_1
00004376 C3                     <1>       retn
                                <1>
                                <1> DISKETTE_IO_1:
                                <1>
00004377 FB                     <1>       STI                         ; INTERRUPTS BACK ON
00004378 55                     <1>       PUSH   eBP                  ; USER REGISTER
00004379 57                     <1>       PUSH   eDI                  ; USER REGISTER
0000437A 52                     <1>       PUSH   eDX                  ; HEAD #, DRIVE # OR USER REGISTER
0000437B 53                     <1>       PUSH   eBX                  ; BUFFER OFFSET PARAMETER OR REGISTER
0000437C 51                     <1>       PUSH   eCX                  ; TRACK #-SECTOR # OR USER REGISTER
0000437D 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
0000437F 06                     <1>       push   es ; 06/02/2015
00004380 1E                     <1>       PUSH   DS                   ; BUFFER SEGMENT PARM OR USER REGISTER
00004381 56                     <1>       PUSH   eSI                  ; USER REGISTERS
                                <1>       ;CALL  DDS                  ; SEGMENT OF BIOS DATA AREA TO DS
                                <1>       ;mov   cx, cs
                                <1>       ;mov   ds, cx
00004382 66B91000               <1>       mov    cx, KDATA
00004386 8ED9                   <1>         mov     ds, cx
00004388 8EC1                   <1>         mov     es, cx
                                <1>
                                <1>       ;CMP   AH,(FNC_TAE-FNC_TAB)/2     ; CHECK FOR > LARGEST FUNCTION
0000438A 80FC19                 <1>       cmp    ah,(FNC_TAE-FNC_TAB)/4  ; 18/02/2015
0000438D 7202                   <1>       JB     short OK_FUNC       ; FUNCTION OK
0000438F B414                   <1>       MOV    AH,14H              ; REPLACE WITH KNOWN INVALID FUNCTION
                                <1> OK_FUNC:
00004391 80FC01                 <1>       CMP    AH,1                ; RESET OR STATUS ?
00004394 760C                   <1>       JBE    short OK_DRV        ; IF RESET OR STATUS DRIVE ALWAYS OK
00004396 80FC08                 <1>       CMP    AH,8                ; READ DRIVE PARMS ?
00004399 7407                   <1>       JZ     short OK_DRV        ; IF SO DRIVE CHECKED LATER
0000439B 80FA01                 <1>       CMP    DL,1                ; DRIVES 0 AND 1 OK
0000439E 7602                   <1>       JBE    short OK_DRV        ; IF 0 OR 1 THEN JUMP
000043A0 B414                   <1>       MOV    AH,14H              ; REPLACE WITH KNOWN INVALID FUNCTION
                                <1> OK_DRV:
000043A2 31C9                   <1>       xor    ecx, ecx
                                <1>       ;mov   esi, ecx ; 08/02/2015
000043A4 89CF                   <1>       mov    edi, ecx ; 08/02/2015
000043A6 88E1                   <1>       MOV    CL,AH               ; CL = FUNCTION
                                <1>       ;XOR   CH,CH               ; CX = FUNCTION
                                <1>       ;SHL   CL, 1               ; FUNCTION TIMES 2
000043A8 C0E102                 <1>       SHL    CL, 2 ; 20/02/2015  ; FUNCTION TIMES 4 (for 32 bit offset)
000043AB BB[E3430000]           <1>       MOV    eBX,FNC_TAB         ; LOAD START OF FUNCTION TABLE
000043B0 01CB                   <1>       ADD    eBX,eCX                  ; ADD OFFSET INTO TABLE => ROUTINE
000043B2 88F4                   <1>       MOV    AH,DH               ; AX = HEAD #,# OF SECTORS OR DASD TYPE
000043B4 30F6                   <1>       XOR    DH,DH               ; DX = DRIVE #
000043B6 6689C6                 <1>       MOV    SI,AX               ; SI = HEAD #,# OF SECTORS OR DASD TYPE
000043B9 6689D7                 <1>       MOV     DI,DX                   ; DI = DRIVE #
                                <1>       ;
                                <1>       ; 11/12/2014
000043BC 8815[DD6D0000]         <1>         mov     [cfd], dl              ; current floppy drive (for 'GET_PARM')
                                <1>       ;
000043C2 8A25[18820100]         <1>       MOV    AH, [DSKETTE_STATUS]      ; LOAD STATUS TO AH FOR STATUS FUNCTION
```

```
000043C8 C605[18820100]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
000043CF FF13                <1>      CALL   dWORD [eBX]        ; CALL THE REQUESTED FUNCTION
000043D1 5E                  <1>      POP    eSI                ; RESTORE ALL REGISTERS
000043D2 1F                  <1>      POP    DS
000043D3 07                  <1>      pop    es     ; 06/02/2015
000043D4 59                  <1>      POP    eCX
000043D5 5B                  <1>      POP    eBX
000043D6 5A                  <1>      POP    eDX
000043D7 5F                  <1>      POP    eDI
000043D8 89E5                <1>      MOV    eBP, eSP
000043DA 50                  <1>      PUSH   eAX
000043DB 9C                  <1>      PUSHFd
000043DC 58                  <1>      POP    eAX
                             <1>      ;MOV   [BP+6], AX
000043DD 89450C              <1>      mov    [ebp+12], eax  ; 18/02/2015, flags
000043E0 58                  <1>      POP    eAX
000043E1 5D                  <1>      POP    eBP
000043E2 CF                  <1>      IRETd
                             <1>
                             <1> ;--------------------------------------------------------------------------------
                             <1> ; DW --> dd (06/02/2015)
000043E3 [47440000]          <1> FNC_TAB      dd     DSK_RESET          ; AH = 00H; RESET
000043E7 [C0440000]          <1>      dd     DSK_STATUS         ; AH = 01H; STATUS
000043EB [D1440000]          <1>      dd     DSK_READ           ; AH = 02H; READ
000043EF [E2440000]          <1>      dd     DSK_WRITE          ; AH = 03H; WRITE
000043F3 [F3440000]          <1>      dd     DSK_VERF           ; AH = 04H; VERIFY
000043F7 [04450000]          <1>      dd     DSK_FORMAT         ; AH = 05H; FORMAT
000043FB [89450000]          <1>      dd     FNC_ERR                 ; AH = 06H; INVALID
000043FF [89450000]          <1>      dd     FNC_ERR                 ; AH = 07H; INVALID
00004403 [96450000]          <1>      dd     DSK_PARMS          ; AH = 08H; READ DRIVE PARAMETERS
00004407 [89450000]          <1>      dd     FNC_ERR                 ; AH = 09H; INVALID
0000440B [89450000]          <1>      dd     FNC_ERR                 ; AH = 0AH; INVALID
0000440F [89450000]          <1>      dd     FNC_ERR                 ; AH = 0BH; INVALID
00004413 [89450000]          <1>      dd     FNC_ERR                 ; AH = 0CH; INVALID
00004417 [89450000]          <1>      dd     FNC_ERR                 ; AH = 0DH; INVALID
0000441B [89450000]          <1>      dd     FNC_ERR                 ; AH = 0EH; INVALID
0000441F [89450000]          <1>      dd     FNC_ERR                 ; AH = 0FH; INVALID
00004423 [89450000]          <1>      dd     FNC_ERR                 ; AH = 10H; INVALID
00004427 [89450000]          <1>      dd     FNC_ERR                 ; AH = 11H; INVALID
0000442B [89450000]          <1>      dd     FNC_ERR                 ; AH = 12H; INVALID
0000442F [89450000]          <1>      dd     FNC_ERR                 ; AH = 13H; INVALID
00004433 [89450000]          <1>      dd     FNC_ERR                 ; AH = 14H; INVALID
00004437 [86460000]          <1>      dd     DSK_TYPE           ; AH = 15H; READ DASD TYPE
0000443B [B6460000]          <1>      dd     DSK_CHANGE         ; AH = 16H; CHANGE STATUS
0000443F [F0460000]          <1>      dd     FORMAT_SET         ; AH = 17H; SET DASD TYPE
00004443 [73470000]          <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:
00004447 66BAF203            <1>      MOV    DX,03F2H           ; ADAPTER CONTROL PORT
0000444B FA                  <1>      CLI                       ; NO INTERRUPTS
0000444C A0[16820100]        <1>      MOV    AL,[MOTOR_STATUS]  ; GET DIGITAL OUTPUT REGISTER REFLECTION
00004451 243F                <1>      AND    AL,00111111B       ; KEEP SELECTED AND MOTOR ON BITS
00004453 C0C004              <1>      ROL    AL,4               ; MOTOR VALUE TO HIGH NIBBLE
                             <1>                                ; DRIVE SELECT TO LOW NIBBLE
00004456 0C08                <1>      OR     AL,00001000B       ; TURN ON INTERRUPT ENABLE
00004458 EE                  <1>      OUT    DX,AL              ; RESET THE ADAPTER
00004459 C605[15820100]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
00004460 E6EB                <2>  out 0ebh,al
                             <1>
                             <1>      ; 17/12/2014
                             <1>      ; AWARD BIOS 1999 - RESETDRIVES (ADISK.ASM)
00004462 B915000000          <1>      mov    ecx, WAITCPU_RESET_ON     ; cx = 21 -- Min. 14 micro seconds !?
                             <1> wdw1:
                             <1>      NEWIODELAY   ; 27/02/2015
00004467 E6EB                <2>  out 0ebh,al
00004469 E2FC                <1>      loop   wdw1
                             <1>      ;
0000446B 0C04                <1>      OR     AL,00000100B       ; TURN OFF RESET BIT
0000446D EE                  <1>      OUT    DX,AL              ; RESET THE ADAPTER
                             <1>      ; 16/12/2014
                             <1>      IODELAY
0000446E EB00                <2>  jmp short $+2
00004470 EB00                <2>  jmp short $+2
                             <1>      ;
```

```
                        <1>       ;STI                    ; ENABLE THE INTERRUPTS
00004472 E8590C0000     <1>       CALL   WAIT_INT         ; WAIT FOR THE INTERRUPT
00004477 723E           <1>       JC     short DR_ERR     ; IF ERROR, RETURN IT
00004479 66B9C000       <1>       MOV    CX,11000000B     ; CL = EXPECTED @NEC_STATUS
                        <1> NXT_DRV:
0000447D 6651           <1>       PUSH   CX               ; SAVE FOR CALL
0000447F B8[B5440000]   <1>       MOV    eAX, DR_POP_ERR  ; LOAD NEC_OUTPUT ERROR ADDRESS
00004484 50             <1>       PUSH   eAX              ; "
00004485 B408           <1>       MOV    AH,08H           ; SENSE INTERRUPT STATUS COMMAND
00004487 E8370B0000     <1>       CALL   NEC_OUTPUT
0000448C 58             <1>       POP    eAX              ; THROW AWAY ERROR RETURN
0000448D E86E0C0000     <1>       CALL   RESULTS                ; READ IN THE RESULTS
00004492 6659           <1>       POP    CX               ; RESTORE AFTER CALL
00004494 7221           <1>       JC     short DR_ERR     ; ERROR RETURN
00004496 3A0D[19820100] <1>       CMP    CL, [NEC_STATUS] ; TEST FOR DRIVE READY TRANSITION
0000449C 7519           <1>       JNZ    short DR_ERR     ; EVERYTHING OK
0000449E FEC1           <1>       INC    CL               ; NEXT EXPECTED @NEC_STATUS
000044A0 80F9C3         <1>       CMP    CL,11000011B     ; ALL POSSIBLE DRIVES CLEARED
000044A3 76D8           <1>       JBE    short NXT_DRV    ; FALL THRU IF 11000100B OR >
                        <1>       ;
000044A5 E886030000     <1>       CALL   SEND_SPEC        ; SEND SPECIFY COMMAND TO NEC
                        <1> RESBAC:
000044AA E83A090000     <1>       CALL   SETUP_END        ; VARIOUS CLEANUPS
000044AF 6689F3         <1>       MOV    BX,SI            ; GET SAVED AL TO BL
000044B2 88D8           <1>       MOV    AL,BL            ; PUT BACK FOR RETURN
000044B4 C3             <1>       RETn
                        <1> DR_POP_ERR:
000044B5 6659           <1>       POP    CX               ; CLEAR STACK
                        <1> DR_ERR:
000044B7 800D[18820100]20 <1>     OR     byte [DSKETTE_STATUS],BAD_NEC ; SET ERROR CODE
000044BE EBEA           <1>       JMP    SHORT RESBAC     ; RETURN FROM RESET
                        <1>
                        <1> ;--------------------------------------------------------------------------
                        <1> ; DISK_STATUS      (AH = 01H)
                        <1> ;     DISKETTE STATUS.
                        <1> ;
                        <1> ; ON ENTRY:  AH : STATUS OF PREVIOUS OPERATION
                        <1> ;
                        <1> ; ON EXIT:   AH, @DSKETTE_STATUS, CY REFLECT STATUS OF PREVIOUS OPERATION.
                        <1> ;--------------------------------------------------------------------------
                        <1> DSK_STATUS:
000044C0 8825[18820100] <1>       MOV    [DSKETTE_STATUS],AH ; PUT BACK FOR SETUP END
000044C6 E81E090000     <1>       CALL   SETUP_END        ; VARIOUS CLEANUPS
000044CB 6689F3         <1>       MOV    BX,SI            ; GET SAVED AL TO BL
000044CE 88D8           <1>       MOV    AL,BL            ; PUT BACK FOR RETURN
000044D0 C3             <1>       RETn
                        <1>
                        <1> ;--------------------------------------------------------------------------
                        <1> ; DISK_READ  (AH = 02H)
                        <1> ;     DISKETTE READ.
                        <1> ;
                        <1> ; ON ENTRY:  DI     : DRIVE #
                        <1> ;            SI-HI  : HEAD #
                        <1> ;            SI-LOW : # OF SECTORS
                        <1> ;            ES     : BUFFER SEGMENT
                        <1> ;            [BP]   : SECTOR #
                        <1> ;            [BP+1] : TRACK #
                        <1> ;            [BP+2] : BUFFER OFFSET
                        <1> ;
                        <1> ; ON EXIT:   @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
                        <1> ;--------------------------------------------------------------------------
                        <1>
                        <1> ; 06/02/2015, ES:BX -> EBX (unix386.s)
                        <1>
                        <1> DSK_READ:
000044D1 8025[16820100]7F <1>     AND    byte [MOTOR_STATUS],01111111B ; INDICATE A READ OPERATION
000044D8 66B846E6       <1>       MOV    AX,0E646H        ; AX = NEC COMMAND, DMA COMMAND
000044DC E859040000     <1>       CALL   RD_WR_VF         ; COMMON READ/WRITE/VERIFY
000044E1 C3             <1>       RETn
                        <1>
                        <1> ;--------------------------------------------------------------------------
                        <1> ; DISK_WRITE (AH = 03H)
                        <1> ;     DISKETTE WRITE.
                        <1> ;
                        <1> ; ON ENTRY:  DI     : DRIVE #
                        <1> ;            SI-HI  : HEAD #
                        <1> ;            SI-LOW : # OF SECTORS
                        <1> ;            ES     : BUFFER SEGMENT
                        <1> ;            [BP]   : SECTOR #
                        <1> ;            [BP+1] : TRACK #
                        <1> ;            [BP+2] : BUFFER OFFSET
                        <1> ;
                        <1> ; ON EXIT:   @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
                        <1> ;--------------------------------------------------------------------------
                        <1>
                        <1> ; 06/02/2015, ES:BX -> EBX (unix386.s)
                        <1>
                        <1> DSK_WRITE:
000044E2 66B84AC5       <1>       MOV    AX,0C54AH          ; AX = NEC COMMAND, DMA COMMAND
000044E6 800D[16820100]80 <1>       OR     byte [MOTOR_STATUS],10000000B ; INDICATE WRITE OPERATION
000044ED E848040000     <1>       CALL   RD_WR_VF         ; COMMON READ/WRITE/VERIFY
000044F2 C3             <1>       RETn
                        <1>
                        <1> ;--------------------------------------------------------------------------
                        <1> ; DISK_VERF  (AH = 04H)
                        <1> ;     DISKETTE VERIFY.
                        <1> ;
                        <1> ; ON ENTRY:  DI     : DRIVE #
                        <1> ;            SI-HI  : HEAD #
                        <1> ;            SI-LOW : # OF SECTORS
                        <1> ;            ES     : BUFFER SEGMENT
                        <1> ;            [BP]   : SECTOR #
                        <1> ;            [BP+1] : TRACK #
                        <1> ;            [BP+2] : BUFFER OFFSET
                        <1> ;
```

```
                            <1> ; ON EXIT:  @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
                            <1> ;--------------------------------------------------------------------------------
                            <1> DSK_VERF:
000044F3 8025[16820100]7F   <1>       AND   byte [MOTOR_STATUS],01111111B ; INDICATE A READ OPERATION
000044FA 66B842E6           <1>       MOV   AX,0E642H           ; AX = NEC COMMAND, DMA COMMAND
000044FE E837040000         <1>       CALL  RD_WR_VF            ; COMMON READ/WRITE/VERIFY
00004503 C3                 <1>       RETn
                            <1>
                            <1> ;--------------------------------------------------------------------------------
                            <1> ; DISK_FORMAT      (AH = 05H)
                            <1> ;     DISKETTE FORMAT.
                            <1> ;
                            <1> ; ON ENTRY: DI     : DRIVE #
                            <1> ;           SI-HI  : HEAD #
                            <1> ;           SI-LOW : # OF SECTORS
                            <1> ;           ES     : BUFFER SEGMENT
                            <1> ;           [BP]   : SECTOR #
                            <1> ;           [BP+1] : TRACK #
                            <1> ;           [BP+2] : BUFFER OFFSET
                            <1> ;           @DISK_POINTER POINTS TO THE PARAMETER TABLE OF THIS DRIVE
                            <1> ;
                            <1> ; ON EXIT:  @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
                            <1> ;--------------------------------------------------------------------------------
                            <1> DSK_FORMAT:
00004504 E870030000         <1>       CALL  XLAT_NEW            ; TRANSLATE STATE TO PRESENT ARCH.
00004509 E86C050000         <1>       CALL  FMT_INIT            ; ESTABLISH STATE IF UNESTABLISHED
0000450E 800D[16820100]80   <1>         OR     byte [MOTOR_STATUS], 10000000B ; INDICATE WRITE OPERATION
00004515 E8B4050000         <1>       CALL  MED_CHANGE          ; CHECK MEDIA CHANGE AND RESET IF SO
0000451A 725D               <1>         JC     short FM_DON           ; MEDIA CHANGED, SKIP
0000451C E80F030000         <1>       CALL  SEND_SPEC           ; SEND SPECIFY COMMAND TO NEC
00004521 E81A060000         <1>       CALL  CHK_LASTRATE        ; ZF=1 ATTEMPT RATE IS SAME AS LAST RATE
00004526 7405               <1>         JZ     short FM_WR            ; YES, SKIP SPECIFY COMMAND
00004528 E8F1050000         <1>       CALL  SEND_RATE           ; SEND DATA RATE TO CONTROLLER
                            <1> FM_WR:
0000452D E8A7060000         <1>       CALL  FMTDMA_SET          ; SET UP THE DMA FOR FORMAT
00004532 7245               <1>         JC     short FM_DON           ; RETURN WITH ERROR
00004534 B44D               <1>       MOV   AH,04DH                ; ESTABLISH THE FORMAT COMMAND
00004536 E804070000         <1>       CALL  NEC_INIT            ; INITIALIZE THE NEC
0000453B 723C               <1>         JC     short FM_DON           ; ERROR - EXIT
0000453D B8[79450000]       <1>         MOV    eAX, FM_DON            ; LOAD ERROR ADDRESS
00004542 50                 <1>       PUSH  eAX                 ; PUSH NEC_OUT ERROR RETURN
00004543 B203               <1>       MOV   DL,3                ; BYTES/SECTOR VALUE TO NEC
00004545 E873090000         <1>       CALL  GET_PARM
0000454A E8740A0000         <1>       CALL  NEC_OUTPUT
0000454F B204               <1>       MOV   DL,4                ; SECTORS/TRACK VALUE TO NEC
00004551 E867090000         <1>       CALL  GET_PARM
00004556 E8680A0000         <1>       CALL  NEC_OUTPUT
0000455B B207               <1>       MOV   DL,7                ; GAP LENGTH VALUE TO NEC
0000455D E85B090000         <1>       CALL  GET_PARM
00004562 E85C0A0000         <1>       CALL  NEC_OUTPUT
00004567 B208               <1>       MOV   DL,8                ; FILLER BYTE TO NEC
00004569 E84F090000         <1>       CALL  GET_PARM
0000456E E8500A0000         <1>       CALL  NEC_OUTPUT
00004573 58                 <1>       POP   eAX                 ; THROW AWAY ERROR
00004574 E844070000         <1>       CALL  NEC_TERM            ; TERMINATE, RECEIVE STATUS, ETC,
                            <1> FM_DON:
00004579 E82C030000         <1>       CALL  XLAT_OLD            ; TRANSLATE STATE TO COMPATIBLE MODE
0000457E E866080000         <1>       CALL  SETUP_END           ; VARIOUS CLEANUPS
00004583 6689F3             <1>       MOV   BX,SI               ; GET SAVED AL TO BL
00004586 88D8               <1>       MOV   AL,BL               ; PUT BACK FOR RETURN
00004588 C3                 <1>       RETn
                            <1>
                            <1> ;--------------------------------------------------------------------------------
                            <1> ; FNC_ERR
                            <1> ;     INVALID FUNCTION REQUESTED OR INVALID DRIVE:
                            <1> ;     SET BAD COMMAND IN STATUS.
                            <1> ;
                            <1> ; ON EXIT:  @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
                            <1> ;--------------------------------------------------------------------------------
                            <1> FNC_ERR:                          ; INVALID FUNCTION REQUEST
00004589 6689F0             <1>       MOV   AX,SI               ; RESTORE AL
0000458C B401               <1>       MOV   AH,BAD_CMD          ; SET BAD COMMAND ERROR
0000458E 8825[18820100]     <1>       MOV   [DSKETTE_STATUS],AH ; STORE IN DATA AREA
00004594 F9                 <1>       STC                       ; SET CARRY INDICATING ERROR
00004595 C3                 <1>       RETn
                            <1>
                            <1> ; 30/08/2020
                            <1> ; 29/08/2020
                            <1> ; 01/06/2016
                            <1> ; 28/05/2016
                            <1> ; 27/05/2016 - TRDOS 386 (TRDOS v.2.0)
                            <1> ;--------------------------------------------------------------------------------
                            <1> ; DISK_PARMS (AH = 08H)
                            <1> ;     READ DRIVE PARAMETERS.
                            <1> ;
                            <1> ; ON ENTRY: DI : DRIVE #
                            <1> ;           ; 27/05/2016
                            <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:
00004596 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
0000459B 29D2           <1>         sub    edx, edx
0000459D 66A1[EE6D0000] <1>         mov     ax, [fd0_type]
000045A3 6621C0         <1>         and     ax, ax
000045A6 0F849B000000   <1>           jz      NON_DRV
000045AC FEC2           <1>         inc     dl
000045AE 20E4           <1>         and     ah, ah
000045B0 7402           <1>         jz      short STO_DL
000045B2 FEC2           <1>         inc     dl
                        <1> STO_DL:
                        <1>         ; 30/08/2020
000045B4 6639FA         <1>         cmp    dx, di
000045B7 0F868A000000   <1>         jna    NON_DRV
                        <1>         ;
                        <1>         ;MOV   [BP+4],DL           ; STORE NUMBER OF DRIVES
000045BD 895508         <1>         mov    [ebp+8], edx ; 20/02/2015
000045C0 6683FF01       <1>         CMP    DI,1                ; CHECK FOR VALID DRIVE
                        <1>         ;JA    short NON_DRV1            ; DRIVE INVALID
000045C4 0F8780000000   <1>         ja     NON_DRV1 ; 29/08/2020
                        <1>         ;MOV   BYTE [BP+5],1       ; MAXIMUM HEAD NUMBER =   1
000045CA C6450901       <1>         mov    byte [ebp+9], 1  ; 20/02/2015
000045CE 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
000045D3 740F           <1>         JZ     short CHK_EST       ; JUMP IF SO
000045D5 E82B020000     <1>         CALL   DR_TYPE_CHECK       ; RTN CS:BX = MEDIA/DRIVE PARAM TBL
000045DA 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
000045DC 8A4B04         <1>         MOV     CL, [eBX+MD.SEC_TRK]     ; GET SECTOR/TRACK
000045DF 8A6B0B         <1>           MOV     CH, [eBX+MD.MAX_TRK]     ; GET MAX. TRACK NUMBER
000045E2 EB36           <1>         JMP    SHORT STO_CX        ; CMOS GOOD, USE CMOS
                        <1> CHK_EST:
000045E4 8AA7[25820100] <1>         MOV    AH, [DSK_STATE+eDI] ; LOAD STATE FOR THIS DRIVE
000045EA F6C410         <1>         TEST   AH,MED_DET          ; CHECK FOR ESTABLISHED STATE
000045ED 745B           <1>         JZ     short NON_DRV1            ; CMOS BAD/INVALID OR UNESTABLISHED
                        <1> USE_EST:
000045EF 80E4C0         <1>         AND    AH,RATE_MSK         ; ISOLATE STATE
000045F2 80FC80         <1>         CMP    AH,RATE_250         ; RATE 250 ?
000045F5 757B           <1>         JNE    short USE_EST2            ; NO, GO CHECK OTHER RATE
                        <1>
                        <1> ;-----      DATA RATE IS 250 KBS, TRY 360 KB TABLE FIRST
                        <1>
000045F7 B001           <1>         MOV    AL,01               ; DRIVE TYPE 1 (360KB)
000045F9 E807020000     <1>         CALL   DR_TYPE_CHECK       ; RTN CS:BX = MEDIA/DRIVE PARAM TBL
000045FE 8A4B04         <1>           MOV     CL, [eBX+MD.SEC_TRK]     ; GET SECTOR/TRACK
00004601 8A6B0B         <1>           MOV     CH, [eBX+MD.MAX_TRK]     ; GET MAX. TRACK NUMBER
00004604 F687[25820100]01 <1>       TEST   byte [DSK_STATE+eDI],TRK_CAPA ; 80 TRACK ?
0000460B 740D           <1>         JZ     short STO_CX        ; MUST BE 360KB DRIVE
                        <1>
                        <1> ;-----      IT IS 1.44 MB DRIVE
                        <1>
                        <1> PARM144:
0000460D B004           <1>         MOV    AL,04               ; DRIVE TYPE 4 (1.44MB)
0000460F E8F1010000     <1>         CALL   DR_TYPE_CHECK       ; RTN CS:BX = MEDIA/DRIVE PARAM TBL
00004614 8A4B04         <1>           MOV     CL, [eBX+MD.SEC_TRK]     ; GET SECTOR/TRACK
00004617 8A6B0B         <1>           MOV     CH, [eBX+MD.MAX_TRK]     ; GET MAX. TRACK NUMBER
                        <1> STO_CX:
0000461A 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>         ;
0000461D 57             <1>         push   edi
0000461E 8B7D04         <1>         mov    edi, [ebp+4]             ; ebx (input), user's buffer address
                        <1>         ; 29/08/2020
00004621 09FF           <1>         or     edi, edi
00004623 7417           <1>         jz     short no_copy_fdpt
                        <1>         ;
00004625 0FB6C0         <1>         movzx  eax, al
00004628 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)
0000462B A3[1C8E0100]   <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>         ;
00004630 89DE           <1>         mov    esi, ebx            ; floppy disk parameter table (16 bytes)
00004632 B910000000     <1>         mov    ecx, 16 ; 16 bytes
00004637 E888CC0000     <1>           call    transfer_to_user_buffer ; trdosk6.s (16/05/2016)
                        <1> no_copy_fdpt:
0000463C 5F             <1>         pop    edi
                        <1> DP_OUT:
0000463D E868020000     <1>         CALL   XLAT_OLD            ; TRANSLATE STATE TO COMPATIBLE MODE
00004642 6631C0         <1>         XOR    AX,AX               ; CLEAR
```

```
00004645 F8                <1>       CLC
00004646 C3                <1>       RETn
                           <1>
                           <1> ;-----      NO DRIYE PRESENT HANDLER
                           <1>
                           <1> NON_DRV:
                           <1>       ;MOV   BYTE [BP+4],0       ; CLEAR NUMBER OF DRIVES
00004647 895508            <1>       mov    [ebp+8], edx ; 0 ; 20/02/2015
                           <1> NON_DRV1:
0000464A 6681FF8000        <1>       CMP    DI,80H              ; CHECK FOR FIXED MEDIA TYPE REQUEST
0000464F 720C              <1>       JB     short NON_DRV2            ; CONTINUE IF NOT REQUEST FALL THROUGH
                           <1>
                           <1> ;-----      FIXED DISK REQUEST FALL THROUGH ERROR
                           <1>
00004651 E854020000        <1>       CALL   XLAT_OLD            ; ELSE TRANSLATE TO COMPATIBLE MODE
00004656 6689F0            <1>       MOV    AX,SI               ; RESTORE AL
00004659 B401              <1>       MOV    AH,BAD_CMD          ; SET BAD COMMAND ERROR
0000465B F9                <1>       STC
0000465C C3                <1>       RETn
                           <1>
                           <1> NON_DRV2:
                           <1>       ;XOR   AX,AX               ; CLEAR PARMS IF NO DRIVES OR CMOS BAD
0000465D 31C0              <1>       xor    eax, eax
0000465F 66894500          <1>       MOV    [eBP],AX            ; TRACKS, SECTORS/TRACK = 0
                           <1>       ;MOV   [BP+5],AH           ; HEAD = 0
00004663 886509            <1>       mov    [ebp+9], ah ; 06/02/2015
                           <1>       ;MOV   [BP+6],AX           ; OFFSET TO DISK_BASE = 0
00004666 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
00004669 E83C020000        <1>       call   XLAT_OLD
                           <1>       ;mov   ah, NOT_RDY ; drive not ready
0000466E B407              <1>       mov    ah, INIT_FAIL ; DRIVE PARAMETER ACTIVITY FAILED
00004670 F9                <1>       stc    ; cf -> 1, ah = 'drive not ready' error code
00004671 C3                <1>       retn
                           <1>
                           <1> ;-----      DATA RATE IS EITHER 300 KBS OR 500 KBS, TRY 1.2 MB TABLE FIRST
                           <1>
                           <1> USE_EST2:
00004672 B002              <1>       MOV    AL,02               ; DRIVE TYPE 2 (1.2MB)
00004674 E88C010000        <1>       CALL   DR_TYPE_CHECK       ; RTN CS:BX = MEDIA/DRIVE PARAM TBL
00004679 8A4B04            <1>         MOV     CL, [eBX+MD.SEC_TRK]    ; GET SECTOR/TRACK
0000467C 8A6B0B            <1>         MOV     CH, [eBX+MD.MAX_TRK]    ; GET MAX. TRACK NUMBER
0000467F 80FC40            <1>       CMP    AH,RATE_300         ; RATE 300 ?
00004682 7496              <1>       JZ     short STO_CX        ; MUST BE 1.2MB DRIVE
00004684 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:
00004686 E8EE010000        <1>       CALL   XLAT_NEW            ; TRANSLATE STATE TO PRESENT ARCH.
0000468B 8A87[25820100]    <1>       MOV    AL,[DSK_STATE+eDI]  ; GET PRESENT STATE INFORMATION
00004691 08C0              <1>       OR     AL,AL               ; CHECK FOR NO DRIVE
00004693 7418              <1>       JZ     short NO_DRV
00004695 B401              <1>       MOV    AH,NOCHGLN          ; NO CHANGE LINE FOR 40 TRACK DRIVE
00004697 A801              <1>       TEST   AL,TRK_CAPA         ; IS THIS DRIVE AN 80 TRACK DRIVE?
00004699 7402              <1>       JZ     short DT_BACK       ; IF NO JUMP
0000469B B402              <1>       MOV    AH,CHGLN            ; CHANGE LINE FOR 80 TRACK DRIVE
                           <1> DT_BACK:
0000469D 6650              <1>       PUSH   AX                  ; SAVE RETURN VALUE
0000469F E806020000        <1>       CALL   XLAT_OLD            ; TRANSLATE STATE TO COMPATIBLE MODE
000046A4 6658              <1>       POP    AX                  ; RESTORE RETURN VALUE
000046A6 F8                <1>       CLC                        ; NO ERROR
000046A7 6689F3            <1>       MOV    BX,SI               ; GET SAVED AL TO BL
000046AA 88D8              <1>       MOV    AL,BL               ; PUT BACK FOR RETURN
000046AC 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
000046AD E8F8010000        <1>       call   XLAT_OLD
000046B2 29C0              <1>       sub    eax, eax
000046B4 F9                <1>       stc    ; cf = 1 -> drive not ready, ah = 0 (disk type = 0)
000046B5 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:
000046B6 E8BE010000        <1>       CALL   XLAT_NEW            ; TRANSLATE STATE TO PRESENT ARCH.
000046BB 8A87[25820100]    <1>       MOV    AL, [DSK_STATE+eDI] ; GET MEDIA STATE INFORMATION
000046C1 08C0              <1>       OR     AL,AL               ; DRIVE PRESENT ?
000046C3 7422              <1>       JZ     short DC_NON        ; JUMP IF NO DRIVE
000046C5 A801              <1>       TEST   AL,TRK_CAPA         ; 80 TRACK DRIVE ?
000046C7 7407              <1>       JZ     short SETIT         ; IF SO , CHECK CHANGE LINE
                           <1> DC0:
000046C9 E88D0A0000        <1>         CALL    READ_DSKCHNG            ; GO CHECK STATE OF DISK CHANGE LINE
```

```
1048 000046CE 7407                <1>       JZ     short FINIS         ; CHANGE LINE NOT ACTIVE
1049                              <1>
1050 000046D0 C605[18820100]06    <1> SETIT:      MOV    byte [DSKETTE_STATUS], MEDIA_CHANGE ; INDICATE MEDIA REMOVED
1051                              <1>
1052 000046D7 E8CE010000          <1> FINIS:      CALL   XLAT_OLD            ; TRANSLATE STATE TO COMPATIBLE MODE
1053 000046DC E808070000          <1>       CALL   SETUP_END           ; VARIOUS CLEANUPS
1054 000046E1 6689F3              <1>       MOV    BX,SI               ; GET SAVED AL TO BL
1055 000046E4 88D8                <1>       MOV    AL,BL               ; PUT BACK FOR RETURN
1056 000046E6 C3                  <1>       RETn
1057                              <1> DC_NON:
1058 000046E7 800D[18820100]80    <1>       OR     byte [DSKETTE_STATUS], TIME_OUT ; SET TIMEOUT, NO DRIVE
1059 000046EE EBE7                <1>       JMP    SHORT FINIS
1060                              <1>
1061                              <1> ;--------------------------------------------------------------------------
1062                              <1> ; FORMAT_SET (AH = 17H)
1063                              <1> ;     THIS ROUTINE IS USED TO ESTABLISH THE TYPE OF MEDIA TO BE USED
1064                              <1> ;     FOR THE FOLLOWING FORMAT OPERATION.
1065                              <1> ;
1066                              <1> ; ON ENTRY:  SI LOW = DASD TYPE FOR FORMAT
1067                              <1> ;            DI     = DRIVE #
1068                              <1> ;
1069                              <1> ; ON EXIT:   @DSKETTE_STATUS REFLECTS STATUS
1070                              <1> ;            AH = @DSKETTE_STATUS
1071                              <1> ;            CY = 1 IF ERROR
1072                              <1> ;--------------------------------------------------------------------------
1073                              <1> FORMAT_SET:
1074 000046F0 E884010000          <1>       CALL   XLAT_NEW            ; TRANSLATE STATE TO PRESENT ARCH.
1075 000046F5 6656                <1>       PUSH   SI                  ; SAVE DASD TYPE
1076 000046F7 6689F0              <1>       MOV    AX,SI               ; AH = ? , AL , DASD TYPE
1077 000046FA 30E4                <1>       XOR    AH,AH               ; AH , 0 , AL , DASD TYPE
1078 000046FC 6689C6              <1>       MOV    SI,AX               ; SI = DASD TYPE
1079 000046FF 80A7[25820100]0F    <1>       AND    byte [DSK_STATE+eDI], ~(MED_DET+DBL_STEP+RATE_MSK) ; CLEAR STATE
1080 00004706 664E                <1>       DEC    SI                  ; CHECK FOR 320/360K MEDIA & DRIVE
1081 00004708 7509                <1>       JNZ    short NOT_320       ; BYPASS IF NOT
1082 0000470A 808F[25820100]90    <1>       OR     byte [DSK_STATE+eDI], MED_DET+RATE_250 ; SET TO 320/360
1083 00004711 EB48                <1>       JMP    SHORT S0
1084                              <1>
1085                              <1> NOT_320:
1086 00004713 E8B6030000          <1>       CALL   MED_CHANGE          ; CHECK FOR TIME_OUT
1087 00004718 803D[18820100]80    <1>       CMP    byte [DSKETTE_STATUS], TIME_OUT
1088 0000471F 743A                <1>       JZ     short S0            ; IF TIME OUT TELL CALLER
1089                              <1> S3:
1090 00004721 664E                <1>       DEC    SI                  ; CHECK FOR 320/360K IN 1.2M DRIVE
1091 00004723 7509                <1>       JNZ    short NOT_320_12    ; BYPASS IF NOT
1092 00004725 808F[25820100]70    <1>       OR     byte [DSK_STATE+eDI], MED_DET+DBL_STEP+RATE_300 ; SET STATE
1093 0000472C EB2D                <1>       JMP    SHORT S0
1094                              <1>
1095                              <1> NOT_320_12:
1096 0000472E 664E                <1>       DEC    SI                  ; CHECK FOR 1.2M MEDIA IN 1.2M DRIVE
1097 00004730 7509                <1>       JNZ    short NOT_12        ; BYPASS IF NOT
1098 00004732 808F[25820100]10    <1>       OR     byte [DSK_STATE+eDI], MED_DET+RATE_500 ; SET STATE VARIABLE
1099 00004739 EB20                <1>       JMP    SHORT S0            ; RETURN TO CALLER
1100                              <1>
1101                              <1> NOT_12:
1102 0000473B 664E                <1>       DEC    SI                  ; CHECK FOR SET DASD TYPE 04
1103 0000473D 752B                <1>       JNZ    short FS_ERR        ; BAD COMMAND EXIT IF NOT VALID TYPE
1104                              <1>
1105 0000473F F687[25820100]04    <1>       TEST   byte [DSK_STATE+eDI], DRV_DET ; DRIVE DETERMINED ?
1106 00004746 740B                <1>       JZ     short ASSUME        ; IF STILL NOT DETERMINED ASSUME
1107 00004748 B050                <1>       MOV    AL,MED_DET+RATE_300
1108 0000474A F687[25820100]02    <1>         TEST    byte [DSK_STATE+eDI], FMT_CAPA ; MULTIPLE FORMAT CAPABILITY ?
1109 00004751 7502                <1>       JNZ    short OR_IT_IN             ; IF 1.2 M THEN DATA RATE 300
1110                              <1>
1111                              <1> ASSUME:
1112 00004753 B090                <1>       MOV    AL,MED_DET+RATE_250 ; SET UP
1113                              <1>
1114                              <1> OR_IT_IN:
1115 00004755 0887[25820100]      <1>       OR     [DSK_STATE+eDI], AL ; OR IN THE CORRECT STATE
1116                              <1> S0:
1117 0000475B E84A010000          <1>       CALL   XLAT_OLD            ; TRANSLATE STATE TO COMPATIBLE MODE
1118 00004760 E884060000          <1>       CALL   SETUP_END           ; VARIOUS CLEANUPS
1119 00004765 665B                <1>       POP    BX                  ; GET SAVED AL TO BL
1120 00004767 88D8                <1>       MOV    AL,BL               ; PUT BACK FOR RETURN
1121 00004769 C3                  <1>       RETn
1122                              <1>
1123                              <1> FS_ERR:
1124 0000476A C605[18820100]01    <1>       MOV    byte [DSKETTE_STATUS], BAD_CMD ; UNKNOWN STATE,BAD COMMAND
1125 00004771 EBE8                <1>       JMP    SHORT S0
1126                              <1>
1127                              <1> ;--------------------------------------------------------------------------
1128                              <1> ; SET_MEDIA  (AH = 18H)
1129                              <1> ;     THIS ROUTINE SETS THE TYPE OF MEDIA AND DATA RATE
1130                              <1> ;     TO BE USED FOR THE FOLLOWING FORMAT OPERATION.
1131                              <1> ;
1132                              <1> ; ON ENTRY:
1133                              <1> ;     [BP]   = SECTOR PER TRACK
1134                              <1> ;     [BP+1] = TRACK #
1135                              <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 00004773 E801010000          <1>       CALL   XLAT_NEW            ; TRANSLATE STATE TO PRESENT ARCH.
1150 00004778 F687[25820100]01    <1>         TEST    byte [DSK_STATE+eDI], TRK_CAPA ; CHECK FOR CHANGE LINE AVAILABLE
1151 0000477F 7415                <1>       JZ     short SM_CMOS       ; JUMP IF 40 TRACK DRIVE
1152 00004781 E848030000          <1>       CALL   MED_CHANGE          ; RESET CHANGE LINE
```

```
1153 00004786 803D[18820100]80    <1>       CMP    byte [DSKETTE_STATUS], TIME_OUT ; IF TIME OUT TELL CALLER
1154 0000478D 746B                <1>       JE     short SM_RTN
1155 0000478F C605[18820100]00    <1>       MOV    byte [DSKETTE_STATUS], 0 ; CLEAR STATUS
1156                              <1> SM_CMOS:
1157 00004796 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 0000479B 745D                <1>       JZ     short SM_RTN        ; RETURN IF SO
1162 0000479D E863000000          <1>       CALL   DR_TYPE_CHECK       ; RTN CS:BX = MEDIA/DRIVE PARAM TBL
1163 000047A2 7231                <1>       JC     short MD_NOT_FND    ; TYPE NOT IN TABLE (BAD CMOS)
1164 000047A4 57                  <1>       PUSH   eDI                 ; SAVE REG.
1165 000047A5 31DB                <1>       XOR    eBX,eBX                  ; BX = INDEX TO DR. TYPE TABLE
1166 000047A7 B906000000          <1>       MOV    eCX,DR_CNT          ; CX = LOOP COUNT
1167                              <1> DR_SEARCH:
1168 000047AC 8AA3[686D0000]      <1>       MOV    AH, [DR_TYPE+eBX]   ; GET DRIVE TYPE
1169 000047B2 80E47F              <1>       AND    AH,BIT7OFF          ; MASK OUT MSB
1170 000047B5 38E0                <1>       CMP    AL,AH               ; DRIVE TYPE MATCH ?
1171 000047B7 7516                <1>       JNE    short NXT_MD        ; NO, CHECK NEXT DRIVE TYPE
1172                              <1> DR_FND:
1173 000047B9 8BBB[696D0000]      <1>       MOV    eDI, [DR_TYPE+eBX+1]      ; DI = MEDIA/DRIVE PARAM TABLE
1174                              <1> MD_SEARCH:
1175 000047BF 8A6704              <1>         MOV     AH, [eDI+MD.SEC_TRK]    ; GET SECTOR/TRACK
1176 000047C2 386500              <1>       CMP    [eBP],AH            ; MATCH?
1177 000047C5 7508                <1>       JNE    short NXT_MD        ; NO, CHECK NEXT MEDIA
1178 000047C7 8A670B              <1>         MOV     AH, [eDI+MD.MAX_TRK]    ; GET MAX. TRACK #
1179 000047CA 386501              <1>       CMP    [eBP+1],AH          ; MATCH?
1180 000047CD 740F                <1>       JE     short MD_FND        ; YES, GO GET RATE
1181                              <1> NXT_MD:
1182                              <1>       ;ADD   BX,3                ; CHECK NEXT DRIVE TYPE
1183 000047CF 83C305              <1>         add  ebx, 5 ; 18/02/2015
1184 000047D2 E2D8                <1>       LOOP    DR_SEARCH
1185 000047D4 5F                  <1>       POP    eDI                 ; RESTORE REG.
1186                              <1> MD_NOT_FND:
1187 000047D5 C605[18820100]0C    <1>       MOV    byte [DSKETTE_STATUS], MED_NOT_FND ; ERROR, MEDIA TYPE NOT FOUND
1188 000047DC EB1C                <1>       JMP    SHORT SM_RTN        ; RETURN
1189                              <1> MD_FND:
1190 000047DE 8A470C              <1>         MOV     AL, [eDI+MD.RATE]       ; GET RATE
1191 000047E1 3C40                <1>       CMP    AL,RATE_300         ; DOUBLE STEP REQUIRED FOR RATE 300
1192 000047E3 7502                <1>       JNE    short MD_SET
1193 000047E5 0C20                <1>       OR     AL,DBL_STEP
1194                              <1> MD_SET:
1195                              <1>       ;MOV   [BP+6],DI           ; SAVE TABLE POINTER IN STACK
1196 000047E7 897D0C              <1>       mov    [ebp+12], edi ; 18/02/2015
1197 000047EA 0C10                <1>       OR     AL,MED_DET          ; SET MEDIA ESTABLISHED
1198 000047EC 5F                  <1>       POP    eDI
1199 000047ED 80A7[25820100]0F    <1>       AND    byte [DSK_STATE+eDI], ~(MED_DET+DBL_STEP+RATE_MSK) ; CLEAR STATE
1200 000047F4 0887[25820100]      <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 000047FA E8AB000000          <1>       CALL   XLAT_OLD            ; TRANSLATE STATE TO COMPATIBLE MODE
1205 000047FF E8E5050000          <1>       CALL   SETUP_END           ; VARIOUS CLEANUPS
1206 00004804 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 00004805 6650                <1>       PUSH   AX
1223 00004807 51                  <1>       PUSH   eCX
1224 00004808 31DB                <1>       XOR    eBX,eBX                  ; BX = INDEX TO DR TYPE TABLE
1225 0000480A B906000000          <1>       MOV    eCX,DR_CNT          ; CX = LOOP COUNT
1226                              <1> TYPE_CHK:
1227 0000480F 8AA3[686D0000]      <1>       MOV    AH,[DR_TYPE+eBX]    ; GET DRIVE TYPE
1228 00004815 38E0                <1>       CMP    AL,AH               ; DRIVE TYPE MATCH?
1229 00004817 740D                <1>       JE     short DR_TYPE_VALID ; YES, RETURN WITH CARRY RESET
1230                              <1>       ;ADD   BX,3                ; CHECK NEXT DRIVE TYPE
1231 00004819 83C305              <1>         add  ebx, 5 ; 16/02/2015 (32 bit address modification)
1232 0000481C E2F1                <1>       LOOP    TYPE_CHK
1233                              <1>       ;
1234 0000481E BB[C76D0000]        <1>       mov    ebx, MD_TBL6        ; 1.44MB fd parameter table
1235                              <1>                                  ; Default for GET_PARM (11/12/2014)
1236                              <1>       ;
1237 00004823 F9                  <1>       STC                        ; DRIVE TYPE NOT FOUND IN TABLE
1238 00004824 EB06                <1>       JMP    SHORT TYPE_RTN
1239                              <1> DR_TYPE_VALID:
1240 00004826 8B9B[696D0000]      <1>       MOV    eBX,[DR_TYPE+eBX+1]       ; BX = MEDIA TABLE
1241                              <1> TYPE_RTN:
1242 0000482C 59                  <1>       POP    eCX
1243 0000482D 6658                <1>       POP    AX
1244 0000482F C3                  <1>       RETn
1245                              <1>
1246                              <1> ;--------------------------------------------------------------
1247                              <1> ; SEND_SPEC                                                  :
1248                              <1> ;     SEND THE SPECIFY COMMAND TO CONTROLLER USING DATA FROM      :
1249                              <1> ;     THE DRIVE PARAMETER TABLE POINTED BY @DISK_POINTER   :
1250                              <1> ; ON ENTRY:  @DISK_POINTER = DRIVE PARAMETER TABLE         :
1251                              <1> ; ON EXIT:   NONE                                    :
1252                              <1> ; REGISTERS ALTERED: CX, DX                                  :
1253                              <1> ;--------------------------------------------------------------
1254                              <1> SEND_SPEC:
1255 00004830 50                  <1>       PUSH   eAX                 ; SAVE AX
1256 00004831 B8[57480000]        <1>       MOV    eAX, SPECBAC        ; LOAD ERROR ADDRESS
1257 00004836 50                  <1>       PUSH   eAX                 ; PUSH NEC_OUT ERROR RETURN
```

```
1258 00004837 B403              <1>       MOV    AH,03H              ; SPECIFY COMMAND
1259 00004839 E885070000        <1>       CALL   NEC_OUTPUT          ; OUTPUT THE COMMAND
1260 0000483E 28D2              <1>       SUB    DL,DL               ; FIRST SPECIFY BYTE
1261 00004840 E878060000        <1>       CALL   GET_PARM            ; GET PARAMETER TO AH
1262 00004845 E879070000        <1>       CALL   NEC_OUTPUT          ; OUTPUT THE COMMAND
1263 0000484A B201              <1>       MOV    DL,1                ; SECOND SPECIFY BYTE
1264 0000484C E86C060000        <1>       CALL   GET_PARM            ; GET PARAMETER TO AH
1265 00004851 E86D070000        <1>       CALL   NEC_OUTPUT          ; OUTPUT THE COMMAND
1266 00004856 58                <1>       POP    eAX                 ; POP ERROR RETURN
1267                            <1> SPECBAC:
1268 00004857 58                <1>       POP    eAX                 ; RESTORE ORIGINAL AX VALUE
1269 00004858 C3                <1>       RETn
1270                            <1>
1271                            <1> ;--------------------------------------------------------------
1272                            <1> ; SEND_SPEC_MD                                               :
1273                            <1> ;     SEND THE SPECIFY COMMAND TO CONTROLLER USING DATA FROM     :
1274                            <1> ;     THE MEDIA/DRIVE PARAMETER TABLE POINTED BY (CS:BX)   :
1275                            <1> ; ON ENTRY:  CS:BX = MEDIA/DRIVE PARAMETER TABLE             :
1276                            <1> ; ON EXIT:   NONE                                      :
1277                            <1> ; REGISTERS ALTERED: AX                                      :
1278                            <1> ;--------------------------------------------------------------
1279                            <1> SEND_SPEC_MD:
1280 00004859 50                <1>       PUSH   eAX                 ; SAVE RATE DATA
1281 0000485A B8[77480000]      <1>       MOV    eAX, SPEC_ESBAC           ; LOAD ERROR ADDRESS
1282 0000485F 50                <1>       PUSH   eAX                 ; PUSH NEC_OUT ERROR RETURN
1283 00004860 B403              <1>       MOV    AH,03H              ; SPECIFY COMMAND
1284 00004862 E85C070000        <1>       CALL   NEC_OUTPUT          ; OUTPUT THE COMMAND
1285 00004867 8A23              <1>         MOV     AH, [eBX+MD.SPEC1]      ; GET 1ST SPECIFY BYTE
1286 00004869 E855070000        <1>       CALL   NEC_OUTPUT          ; OUTPUT THE COMMAND
1287 0000486E 8A6301            <1>         MOV     AH, [eBX+MD.SPEC2]      ; GET SECOND SPECIFY BYTE
1288 00004871 E84D070000        <1>       CALL   NEC_OUTPUT          ; OUTPUT THE COMMAND
1289 00004876 58                <1>       POP    eAX                 ; POP ERROR RETURN
1290                            <1> SPEC_ESBAC:
1291 00004877 58                <1>       POP    eAX                 ; RESTORE ORIGINAL AX VALUE
1292 00004878 C3                <1>       RETn
1293                            <1>
1294                            <1> ;-----------------------------------------------------------------------
1295                            <1> ; XLAT_NEW
1296                            <1> ;     TRANSLATES DISKETTE STATE LOCATIONS FROM COMPATIBLE
1297                            <1> ;     MODE TO NEW ARCHITECTURE.
1298                            <1> ;
1299                            <1> ; ON ENTRY:  DI = DRIVE #
1300                            <1> ;-----------------------------------------------------------------------
1301                            <1> XLAT_NEW:
1302 00004879 83FF01            <1>       CMP    eDI,1                   ; VALID DRIVE
1303 0000487C 7725              <1>       JA     short XN_OUT            ; IF INVALID BACK
1304 0000487E 80BF[25820100]00  <1>       CMP    byte [DSK_STATE+eDI], 0         ; NO DRIVE ?
1305 00004885 741D              <1>       JZ     short DO_DET            ; IF NO DRIVE ATTEMPT DETERMINE
1306 00004887 6689F9            <1>       MOV    CX,DI                   ; CX = DRIVE NUMBER
1307 0000488A C0E102            <1>       SHL    CL,2                    ; CL = SHIFT COUNT, A=0, B=4
1308 0000488D A0[24820100]      <1>       MOV    AL, [HF_CNTRL]                ; DRIVE INFORMATION
1309 00004892 D2C8              <1>       ROR    AL,CL                   ; TO LOW NIBBLE
1310 00004894 2407              <1>       AND    AL,DRV_DET+FMT_CAPA+TRK_CAPA     ; KEEP DRIVE BITS
1311 00004896 80A7[25820100]F8  <1>         AND     byte [DSK_STATE+eDI], ~(DRV_DET+FMT_CAPA+TRK_CAPA)
1312 0000489D 0887[25820100]    <1>       OR     [DSK_STATE+eDI], AL      ; UPDATE DRIVE STATE
1313                            <1> XN_OUT:
1314 000048A3 C3                <1>       RETn
1315                            <1> DO_DET:
1316 000048A4 E8BF080000        <1>       CALL   DRIVE_DET               ; TRY TO DETERMINE
1317 000048A9 C3                <1>       RETn
1318                            <1>
1319                            <1> ;-----------------------------------------------------------------------
1320                            <1> ; XLAT_OLD
1321                            <1> ;     TRANSLATES DISKETTE STATE LOCATIONS FROM NEW
1322                            <1> ;     ARCHITECTURE TO COMPATIBLE MODE.
1323                            <1> ;
1324                            <1> ; ON ENTRY:  DI = DRIVE
1325                            <1> ;-----------------------------------------------------------------------
1326                            <1> XLAT_OLD:
1327 000048AA 83FF01            <1>       CMP    eDI,1              ; VALID DRIVE ?
1328                            <1>         ;JA     short XO_OUT           ; IF INVALID BACK
1329 000048AD 0F8786000000      <1>         ja      XO_OUT
1330 000048B3 80BF[25820100]00  <1>         CMP  byte [DSK_STATE+eDI],0    ; NO DRIVE ?
1331 000048BA 747D              <1>       JZ     short XO_OUT       ; IF NO DRIVE TRANSLATE DONE
1332                            <1>
1333                            <1> ;-----      TEST FOR SAVED DRIVE INFORMATION ALREADY SET
1334                            <1>
1335 000048BC 6689F9            <1>       MOV    CX,DI              ; CX = DRIVE NUMBER
1336 000048BF C0E102            <1>       SHL    CL,2               ; CL = SHIFT COUNT, A=0, B=4
1337 000048C2 B402              <1>       MOV    AH,FMT_CAPA        ; LOAD MULTIPLE DATA RATE BIT MASK
1338 000048C4 D2CC              <1>       ROR    AH,CL              ; ROTATE BY MASK
1339 000048C6 8425[24820100]    <1>       TEST   [HF_CNTRL], AH           ; MULTIPLE-DATA RATE DETERMINED ?
1340 000048CC 751C              <1>       JNZ    short SAVE_SET           ; IF SO, NO NEED TO RE-SAVE
1341                            <1>
1342                            <1> ;-----      ERASE DRIVE BITS IN @HF_CNTRL FOR THIS DRIVE
1343                            <1>
1344 000048CE B407              <1>       MOV    AH,DRV_DET+FMT_CAPA+TRK_CAPA ; MASK TO KEEP
1345 000048D0 D2CC              <1>       ROR    AH,CL              ; FIX MASK TO KEEP
1346 000048D2 F6D4              <1>       NOT    AH                 ; TRANSLATE MASK
1347 000048D4 2025[24820100]    <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 000048DA 8A87[25820100]    <1>       MOV    AL, [DSK_STATE+eDI] ; ACCESS STATE
1352 000048E0 2407              <1>       AND    AL,DRV_DET+FMT_CAPA+TRK_CAPA ; KEEP DRIVE BITS
1353 000048E2 D2C8              <1>       ROR    AL,CL              ; FIX FOR THIS DRIVE
1354 000048E4 0805[24820100]    <1>       OR     [HF_CNTRL], AL           ; UPDATE SAVED DRIVE STATE
1355                            <1>
1356                            <1> ;-----      TRANSLATE TO COMPATIBILITY MODE
1357                            <1>
1358                            <1> SAVE_SET:
1359 000048EA 8AA7[25820100]    <1>       MOV    AH, [DSK_STATE+eDI] ; ACCESS STATE
1360 000048F0 88E7              <1>       MOV    BH,AH              ; TO BH FOR LATER
1361 000048F2 80E4C0            <1>       AND    AH,RATE_MSK        ; KEEP ONLY RATE
1362 000048F5 80FC00            <1>       CMP    AH,RATE_500        ; RATE 500 ?
```

```
1363 000048F8 7410                <1>       JZ     short CHK_144       ; YES 1.2/1.2 OR 1.44/1.44
1364 000048FA B001                <1>       MOV    AL,M3D1U            ; AL = 360 IN 1.2 UNESTABLISHED
1365 000048FC 80FC40              <1>       CMP    AH,RATE_300         ; RATE 300 ?
1366 000048FF 7518                <1>       JNZ    short CHK_250       ; NO, 360/360, 720/720 OR 720/1.44
1367 00004901 F6C720              <1>       TEST   BH,DBL_STEP         ; CHECK FOR DOUBLE STEP
1368 00004904 751F                <1>       JNZ    short TST_DET       ; MUST BE 360 IN 1.2
1369                              <1> UNKNO:
1370 00004906 B007                <1>       MOV    AL,MED_UNK          ; NONE OF THE ABOVE
1371 00004908 EB22                <1>       JMP    SHORT AL_SET        ; PROCESS COMPLETE
1372                              <1> CHK_144:
1373 0000490A 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 0000490F 74F5                <1>       jz     short UNKNO ;; 20/02/2015
1377 00004911 3C02                <1>       CMP    AL,2                ; 1.2MB DRIVE ?
1378 00004913 75F1                <1>       JNE    short UNKNO         ; NO, GO SET 'NONE OF ABOVE'
1379 00004915 B002                <1>       MOV    AL,M1D1U            ; AL = 1.2 IN 1.2 UNESTABLISHED
1380 00004917 EB0C                <1>       JMP    SHORT TST_DET
1381                              <1> CHK_250:
1382 00004919 B000                <1>       MOV    AL,M3D3U            ; AL = 360 IN 360 UNESTABLISHED
1383 0000491B 80FC80              <1>       CMP    AH,RATE_250         ; RATE 250 ?
1384 0000491E 75E6                <1>       JNZ    short UNKNO         ; IF SO FALL IHRU
1385 00004920 F6C701              <1>       TEST   BH,TRK_CAPA         ; 80 TRACK CAPABILITY ?
1386 00004923 75E1                <1>       JNZ    short UNKNO         ; IF SO JUMP, FALL THRU TEST DET
1387                              <1> TST_DET:
1388 00004925 F6C710              <1>       TEST   BH,MED_DET          ; DETERMINED ?
1389 00004928 7402                <1>       JZ     short AL_SET        ; IF NOT THEN SET
1390 0000492A 0403                <1>       ADD    AL,3                ; MAKE DETERMINED/ESTABLISHED
1391                              <1> AL_SET:
1392 0000492C 80A7[25820100]F8    <1>       AND    byte [DSK_STATE+eDI], ~(DRV_DET+FMT_CAPA+TRK_CAPA) ; CLEAR DRIVE
1393 00004933 0887[25820100]      <1>       OR     [DSK_STATE+eDI], AL ; REPLACE WITH COMPATIBLE MODE
1394                              <1> XO_OUT:
1395 00004939 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 0000493A 6650                <1>       PUSH   AX                  ; SAVE DMA, NEC PARAMETERS
1409 0000493C E838FFFFFF          <1>       CALL   XLAT_NEW            ; TRANSLATE STATE TO PRESENT ARCH.
1410 00004941 E8F3000000          <1>       CALL   SETUP_STATE         ; INITIALIZE START AND END RATE
1411 00004946 6658                <1>       POP    AX                  ; RESTORE READ/WRITE/VERIFY
1412                              <1> DO_AGAIN:
1413 00004948 6650                <1>       PUSH   AX                  ; SAVE READ/WRITE/VERIFY PARAMETER
1414 0000494A E87F010000          <1>       CALL   MED_CHANGE          ; MEDIA CHANGE AND RESET IF CHANGED
1415 0000494F 6658                <1>       POP    AX                  ; RESTORE READ/WRITE/VERIFY
1416 00004951 0F82C9000000        <1>         JC     RWV_END                 ; MEDIA CHANGE ERROR OR TIME-OUT
1417                              <1> RWV:
1418 00004957 6650                <1>       PUSH   AX                  ; SAVE READ/WRITE/VERIFY PARAMETER
1419 00004959 8AB7[25820100]      <1>       MOV    DH, [DSK_STATE+eDI] ; GET RATE STATE OF THIS DRIVE
1420 0000495F 80E6C0              <1>       AND    DH,RATE_MSK         ; KEEP ONLY RATE
1421 00004962 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 00004967 7451                <1>       jz     short RWV_ASSUME ; 20/02/2015
1425 00004969 3C01                <1>       CMP    AL,1                ; 40 TRACK DRIVE?
1426 0000496B 750D                <1>       JNE    short RWV_1         ; NO, BYPASS CMOS VALIDITY CHECK
1427 0000496D F687[25820100]01    <1>       TEST   byte [DSK_STATE+eDI], TRK_CAPA ; CHECK FOR 40 TRACK DRIVE
1428 00004974 7413                <1>       JZ     short RWV_2         ; YES, CMOS IS CORRECT
1429 00004976 B002                <1>       MOV    AL,2                ; CHANGE TO 1.2M
1430 00004978 EB0F                <1>       JMP    SHORT RWV_2
1431                              <1> RWV_1:
1432 0000497A 720D                <1>       JB     short RWV_2         ; NO DRIVE SPECIFIED, CONTINUE
1433 0000497C F687[25820100]01    <1>       TEST    byte [DSK_STATE+eDI], TRK_CAPA ; IS IT REALLY 40 TRACK?
1434 00004983 7504                <1>       JNZ    short RWV_2         ; NO, 80 TRACK
1435 00004985 B001                <1>       MOV    AL,1                ; IT IS 40 TRACK, FIX CMOS VALUE
1436 00004987 EB04                <1>       jmp    short rwv_3
1437                              <1> RWV_2:
1438 00004989 08C0                <1>       OR     AL,AL               ; TEST FOR NO DRIVE
1439 0000498B 742D                <1>       JZ     short RWV_ASSUME    ; ASSUME TYPE, USE MAX TRACK
1440                              <1> rwv_3:
1441 0000498D E873FEFFFF          <1>       CALL   DR_TYPE_CHECK       ; RTN CS:BX = MEDIA/DRIVE PARAM TBL.
1442 00004992 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 00004994 57                  <1>       PUSH   eDI                 ; SAVE DRIVE #
1447 00004995 31DB                <1>       XOR    eBX,eBX                   ; BX = INDEX TO DR_TYPE TABLE
1448 00004997 B906000000          <1>       MOV    eCX,DR_CNT          ; CX = LOOP COUNT
1449                              <1> RWV_DR_SEARCH:
1450 0000499C 8AA3[686D0000]      <1>       MOV    AH, [DR_TYPE+eBX]   ; GET DRIVE TYPE
1451 000049A2 80E47F              <1>       AND    AH,BIT7OFF          ; MASK OUT MSB
1452 000049A5 38E0                <1>       CMP    AL,AH               ; DRIVE TYPE MATCH?
1453 000049A7 750B                <1>       JNE    short RWV_NXT_MD    ; NO, CHECK NEXT DRIVE TYPE
1454                              <1> RWV_DR_FND:
1455 000049A9 8BBB[696D0000]      <1>       MOV    eDI, [DR_TYPE+eBX+1]      ; DI = MEDIA/DRIVE PARAMETER TABLE
1456                              <1> RWV_MD_SEARH:
1457 000049AF 3A770C              <1>         CMP     DH, [eDI+MD.RATE]       ; MATCH?
1458 000049B2 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 000049B4 83C305              <1>       add    eBX, 5
1462 000049B7 E2E3                <1>       LOOP   RWV_DR_SEARCH
1463 000049B9 5F                  <1>       POP    eDI                 ; RESTORE DRIVE #
1464                              <1>
1465                              <1> ;-----       ASSUME PRIMARY DRIVE IS INSTALLED AS SHIPPED
1466                              <1>
1467                              <1> RWV_ASSUME:
```

```
1468 000049BA BB[866D0000]        <1>       MOV    eBX, MD_TBL1        ; POINT TO 40 TRACK 250 KBS
1469 000049BF F687[25820100]01    <1>       TEST   byte [DSK_STATE+eDI], TRK_CAPA ; TEST FOR 80 TRACK
1470 000049C6 740A                <1>       JZ     short RWV_MD_FND1   ; MUST BE 40 TRACK
1471 000049C8 BB[A06D0000]        <1>       MOV    eBX, MD_TBL3        ; POINT TO 80 TRACK 500 KBS
1472 000049CD 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 000049CF 89FB                <1>       MOV    eBX,eDI                     ; BX = MEDIA/DRIVE PARAMETER TABLE
1478 000049D1 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 000049D2 E882FEFFFF          <1>       CALL   SEND_SPEC_MD
1484 000049D7 E864010000          <1>       CALL   CHK_LASTRATE        ; ZF=1 ATTEMP RATE IS SAME AS LAST RATE
1485 000049DC 7405                <1>       JZ     short RWV_DBL       ; YES,SKIP SEND RATE COMMAND
1486 000049DE E83B010000          <1>       CALL   SEND_RATE           ; SEND DATA RATE TO NEC
1487                              <1> RWV_DBL:
1488 000049E3 53                  <1>       PUSH   eBX                 ; SAVE MEDIA/DRIVE PARAM TBL ADDRESS
1489 000049E4 E822040000          <1>       CALL   SETUP_DBL           ; CHECK FOR DOUBLE STEP
1490 000049E9 5B                  <1>       POP    eBX                 ; RESTORE ADDRESS
1491 000049EA 7226                <1>       JC     short CHK_RET       ; ERROR FROM READ ID, POSSIBLE RETRY
1492 000049EC 6658                <1>       POP    AX                  ; RESTORE NEC, DMA COMMAND
1493 000049EE 6650                <1>       PUSH   AX                  ; SAVE NEC COMMAND
1494 000049F0 53                  <1>       PUSH   eBX                 ; SAVE MEDIA/DRIVE PARAM TBL ADDRESS
1495 000049F1 E861010000          <1>       CALL   DMA_SETUP           ; SET UP THE DMA
1496 000049F6 5B                  <1>       POP    eBX
1497 000049F7 6658                <1>       POP    AX                  ; RESTORE NEC COMMAND
1498 000049F9 722F                <1>       JC     short RWV_BAC       ; CHECK FOR DMA BOUNDARY ERROR
1499 000049FB 6650                <1>       PUSH   AX                  ; SAVE NEC COMMAND
1500 000049FD 53                  <1>       PUSH   eBX                 ; SAVE MEDIA/DRIVE PARAM TBL ADDRESS
1501 000049FE E83C020000          <1>       CALL   NEC_INIT            ; INITIALIZE NEC
1502 00004A03 5B                  <1>       POP    eBX                 ; RESTORE ADDRESS
1503 00004A04 720C                <1>       JC     short CHK_RET       ; ERROR - EXIT
1504 00004A06 E866020000          <1>       CALL   RWV_COM                   ; OP CODE COMMON TO READ/WRITE/VERIFY
1505 00004A0B 7205                <1>       JC     short CHK_RET       ; ERROR - EXIT
1506 00004A0D E8AB020000          <1>       CALL   NEC_TERM            ; TERMINATE, GET STATUS, ETC.
1507                              <1> CHK_RET:
1508 00004A12 E84A030000          <1>       CALL   RETRY               ; CHECK FOR, SETUP RETRY
1509 00004A17 6658                <1>       POP    AX                  ; RESTORE READ/WRITE/VERIFY PARAMETER
1510 00004A19 7305                <1>       JNC    short RWV_END       ; CY = 0 NO RETRY
1511 00004A1B E928FFFFFF          <1>         JMP     DO_AGAIN                ; CY = 1 MEANS RETRY
1512                              <1> RWV_END:
1513 00004A20 E8F4020000          <1>       CALL   DSTATE              ; ESTABLISH STATE IF SUCCESSFUL
1514 00004A25 E887030000          <1>       CALL   NUM_TRANS           ; AL = NUMBER TRANSFERRED
1515                              <1> RWV_BAC:                         ; BAD DMA ERROR ENTRY
1516 00004A2A 6650                <1>       PUSH   AX                  ; SAVE NUMBER TRANSFERRED
1517 00004A2C E879FEFFFF          <1>       CALL   XLAT_OLD            ; TRANSLATE STATE TO COMPATIBLE MODE
1518 00004A31 6658                <1>       POP    AX                  ; RESTORE NUMBER TRANSFERRED
1519 00004A33 E8B1030000          <1>       CALL   SETUP_END           ; VARIOUS CLEANUPS
1520 00004A38 C3                  <1>       RETn
1521                              <1>
1522                              <1> ;--------------------------------------------------------------------------
1523                              <1> ; SETUP_STATE:     INITIALIZES START AND END RATES.
1524                              <1> ;--------------------------------------------------------------------------
1525                              <1> SETUP_STATE:
1526 00004A39 F687[25820100]10    <1>       TEST   byte [DSK_STATE+eDI], MED_DET ; MEDIA DETERMINED ?
1527 00004A40 7537                <1>       JNZ    short J1C           ; NO STATES IF DETERMINED
1528 00004A42 66B84000            <1>         MOV     AX,(RATE_500*256)+RATE_300  ; AH = START RATE, AL = END RATE
1529 00004A46 F687[25820100]04    <1>       TEST   byte [DSK_STATE+eDI],DRV_DET ; DRIVE ?
1530 00004A4D 740D                <1>       JZ     short AX_SET        ; DO NOT KNOW DRIVE
1531 00004A4F F687[25820100]02    <1>       TEST   byte [DSK_STATE+eDI], FMT_CAPA ; MULTI-RATE?
1532 00004A56 7504                <1>       JNZ    short AX_SET        ; JUMP IF YES
1533 00004A58 66B88080            <1>         MOV     AX,RATE_250*257         ; START A END RATE 250 FOR 360 DRIVE
1534                              <1> AX_SET:
1535 00004A5C 80A7[25820100]1F    <1>       AND    byte [DSK_STATE+eDI], ~(RATE_MSK+DBL_STEP) ; TURN OFF THE RATE
1536 00004A63 08A7[25820100]      <1>       OR     [DSK_STATE+eDI], AH ; RATE FIRST TO TRY
1537 00004A69 8025[20820100]F3    <1>       AND    byte [LASTRATE], ~STRT_MSK ; ERASE LAST TO TRY RATE BITS
1538 00004A70 C0C804              <1>       ROR    AL,4                ; TO OPERATION LAST RATE LOCATION
1539 00004A73 0805[20820100]      <1>       OR     [LASTRATE], AL             ; LAST RATE
1540                              <1> J1C:
1541 00004A79 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 00004A7A F687[25820100]10    <1>       TEST   byte [DSK_STATE+eDI], MED_DET ; IS MEDIA ESTABLISHED
1548 00004A81 7546                <1>       JNZ    short F1_OUT        ; IF SO RETURN
1549 00004A83 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 00004A88 7440                <1>       jz     short CL_DRV ;; 20/02/2015
1553 00004A8A FEC8                <1>       DEC    AL                  ; MAKE ZERO ORIGIN
1554                              <1>       ;;JS   short CL_DRV        ; NO DRIVE IF AL 0
1555 00004A8C 8AA7[25820100]      <1>       MOV    AH, [DSK_STATE+eDI] ; AH = CURRENT STATE
1556 00004A92 80E40F              <1>       AND    AH, ~(MED_DET+DBL_STEP+RATE_MSK) ; CLEAR
1557 00004A95 08C0                <1>       OR     AL,AL               ; CHECK FOR 360
1558 00004A97 7505                <1>       JNZ    short N_360         ; IF 360 WILL BE 0
1559 00004A99 80CC90              <1>       OR     AH,MED_DET+RATE_250 ; ESTABLISH MEDIA
1560 00004A9C EB25                <1>       JMP    SHORT SKP_STATE           ; SKIP OTHER STATE PROCESSING
1561                              <1> N_360:
1562 00004A9E FEC8                <1>       DEC    AL                  ; 1.2 M DRIVE
1563 00004AA0 7505                <1>       JNZ    short N_12          ; JUMP IF NOT
1564                              <1> F1_RATE:
1565 00004AA2 80CC10              <1>       OR     AH,MED_DET+RATE_500 ; SET FORMAT RATE
1566 00004AA5 EB1C                <1>       JMP    SHORT SKP_STATE           ; SKIP OTHER STATE PROCESSING
1567                              <1> N_12:
1568 00004AA7 FEC8                <1>       DEC    AL                  ; CHECK FOR TYPE 3
1569 00004AA9 750F                <1>       JNZ    short N_720         ; JUMP IF NOT
1570 00004AAB F6C404              <1>       TEST   AH,DRV_DET          ; IS DRIVE DETERMINED
1571 00004AAE 7410                <1>       JZ     short ISNT_12       ; TREAT AS NON 1.2 DRIVE
1572 00004AB0 F6C402              <1>       TEST   AH,FMT_CAPA         ; IS 1.2M
```

```
1573 00004AB3 740B                <1>       JZ     short ISNT_12       ; JUMP IF NOT
1574 00004AB5 80CC50              <1>       OR     AH,MED_DET+RATE_300 ; RATE 300
1575 00004AB8 EB09                <1>       JMP    SHORT SKP_STATE           ; CONTINUE
1576                              <1> N_720:
1577 00004ABA FEC8                <1>       DEC    AL                  ; CHECK FOR TYPE 4
1578 00004ABC 750C                <1>       JNZ    short CL_DRV        ; NO DRIVE, CMOS BAD
1579 00004ABE EBE2                <1>       JMP    SHORT F1_RATE
1580                              <1> ISNT_12:
1581 00004AC0 80CC90              <1>       OR     AH,MED_DET+RATE_250 ; MUST BE RATE 250
1582                              <1>
1583                              <1> SKP_STATE:
1584 00004AC3 88A7[25820100]      <1>       MOV    [DSK_STATE+eDI], AH ; STORE AWAY
1585                              <1> F1_OUT:
1586 00004AC9 C3                  <1>       RETn
1587                              <1> CL_DRV:
1588 00004ACA 30E4                <1>       XOR    AH,AH               ; CLEAR STATE
1589 00004ACC 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 00004ACE E888060000          <1>       CALL   READ_DSKCHNG        ; READ DISK CHANCE LINE STATE
1601 00004AD3 7447                <1>       JZ     short MC_OUT        ; BYPASS HANDLING DISK CHANGE LINE
1602 00004AD5 80A7[25820100]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 00004ADC 6689F9              <1>       MOV    CX,DI               ; CL = DRIVE 0
1609 00004ADF B001                <1>       MOV    AL,1                ; MOTOR ON BIT MASK
1610 00004AE1 D2E0                <1>       SHL    AL,CL               ; TO APPROPRIATE POSITION
1611 00004AE3 F6D0                <1>       NOT    AL                  ; KEEP ALL BUT MOTOR ON
1612 00004AE5 FA                  <1>       CLI                        ; NO INTERRUPTS
1613 00004AE6 2005[16820100]      <1>       AND    [MOTOR_STATUS], AL  ; TURN MOTOR OFF INDICATOR
1614 00004AEC FB                  <1>       STI                        ; INTERRUPTS ENABLED
1615 00004AED 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 00004AF2 E850F9FFFF          <1>       CALL   DSK_RESET           ; RESET NEC
1620 00004AF7 B501                <1>       MOV    CH,01H              ; MOVE TO CYLINDER 1
1621 00004AF9 E8FF040000          <1>       CALL   SEEK                ; ISSUE SEEK
1622 00004AFE 30ED                <1>       XOR    CH,CH               ; MOVE TO CYLINDER 0
1623 00004B00 E8F8040000          <1>       CALL   SEEK                ; ISSUE SEEK
1624 00004B05 C605[18820100]06    <1>       MOV    byte [DSKETTE_STATUS], MEDIA_CHANGE ; STORE IN STATUS
1625                              <1> OK1:
1626 00004B0C E84A060000          <1>       CALL   READ_DSKCHNG        ; CHECK MEDIA CHANGED AGAIN
1627 00004B11 7407                <1>       JZ     short OK2           ; IF ACTIVE, NO DISKETTE, TIMEOUT
1628                              <1> OK4:
1629 00004B13 C605[18820100]80    <1>       MOV    byte [DSKETTE_STATUS], TIME_OUT ; TIMEOUT IF DRIVE EMPTY
1630                              <1> OK2:
1631 00004B1A F9                  <1>       STC                        ; MEDIA CHANGED, SET CY
1632 00004B1B C3                  <1>       RETn
1633                              <1> MC_OUT:
1634 00004B1C F8                  <1>       CLC                        ; NO MEDIA CHANGED, CLEAR CY
1635 00004B1D 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 00004B1E 6650                <1>       PUSH   AX                  ; SAVE REG.
1646 00004B20 8025[20820100]3F    <1>       AND    byte [LASTRATE], ~SEND_MSK ; ELSE CLEAR LAST RATE ATTEMPTED
1647 00004B27 8A87[25820100]      <1>       MOV    AL, [DSK_STATE+eDI] ; GET RATE STATE OF THIS DRIVE
1648 00004B2D 24C0                <1>       AND    AL,SEND_MSK         ; KEEP ONLY RATE BITS
1649 00004B2F 0805[20820100]      <1>       OR     [LASTRATE], AL            ; SAVE NEW RATE FOR NEXT CHECK
1650 00004B35 C0C002              <1>       ROL    AL,2                ; MOVE TO BIT OUTPUT POSITIONS
1651 00004B38 66BAF703            <1>       MOV    DX,03F7H            ; OUTPUT NEW DATA RATE
1652 00004B3C EE                  <1>       OUT    DX,AL
1653 00004B3D 6658                <1>       POP    AX                  ; RESTORE REG.
1654 00004B3F 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 #
1661                              <1> ; ON EXIT:
1662                              <1> ;     ZF =  1 DATA RATE IS THE SAME AS THE LAST RATE SENT TO NEC
1663                              <1> ;     ZF =  0 DATA RATE IS DIFFERENT FROM LAST RATE
1664                              <1> ; REGISTERS ALTERED: DX
1665                              <1> ;--------------------------------------------------------------------------------
1666                              <1> CHK_LASTRATE:
1667 00004B40 6650                <1>       PUSH   AX                  ; SAVE REG
1668 00004B42 2225[20820100]      <1>       AND    AH, [LASTRATE]            ; GET LAST DATA RATE SELECTED
1669 00004B48 8A87[25820100]      <1>       MOV    AL, [DSK_STATE+eDI] ; GET RATE STATE OF THIS DRIVE
1670 00004B4E 6625C0C0            <1>         AND     AX, SEND_MSK*257        ; KEEP ONLY RATE BITS OF BOTH
1671 00004B52 38E0                <1>       CMP    AL, AH              ; COMPARE TO PREVIOUSLY TRIED
1672                              <1>                                  ; ZF = 1 RATE IS THE SAME
1673 00004B54 6658                <1>       POP    AX                  ; RESTORE REG.
1674 00004B56 C3                  <1>       RETn
1675                              <1>
1676                              <1> ;--------------------------------------------------------------------------------
1677                              <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
00004B57 8B5504         <1>       mov   edx, [ebp+4]        ; Buffer address
00004B5A F7C2000000FF   <1>       test  edx, 0FF000000h           ; 16 MB limit (22/08/2015, bugfix)
00004B60 756E           <1>       jnz   short dma_bnd_err_stc
                        <1>       ;
00004B62 6650           <1>       push  ax                  ; DMA command
00004B64 52             <1>       push  edx                 ; *
00004B65 B203           <1>       mov   dl, 3               ; GET BYTES/SECTOR PARAMETER
00004B67 E851030000     <1>       call  GET_PARM            ;
00004B6C 88E1           <1>       mov   cl, ah                    ; SHIFT COUNT (0=128, 1=256, 2=512 ETC)
00004B6E 6689F0         <1>       mov   ax, si              ; Sector count
00004B71 88C4           <1>       mov   ah, al              ; AH =  # OF SECTORS
00004B73 28C0           <1>       sub   al, al              ; AL = 0, AX = # SECTORS * 256
00004B75 66D1E8         <1>       shr   ax, 1               ; AX = # SECTORS * 128
00004B78 66D3E0         <1>       shl   ax, cl              ; SHIFT BY PARAMETER VALUE
00004B7B 6648           <1>       dec   ax                  ; -1 FOR DMA VALUE
00004B7D 6689C1         <1>       mov   cx, ax
00004B80 5A             <1>       pop   edx                 ; *
00004B81 6658           <1>       pop   ax
00004B83 3C42           <1>       cmp   al, 42h
00004B85 7507           <1>         jne     short NOT_VERF
00004B87 BA0000FF00     <1>       mov   edx, 0FF0000h
00004B8C EB08           <1>       jmp   short J33
                        <1> NOT_VERF:
00004B8E 6601CA         <1>       add   dx, cx              ; check for overflow
00004B91 723E           <1>       jc    short dma_bnd_err
                        <1>       ;
00004B93 6629CA         <1>       sub   dx, cx              ; Restore start address
                        <1> J33:
00004B96 FA             <1>       CLI                       ; DISABLE INTERRUPTS DURING DMA SET-UP
00004B97 E60C           <1>       OUT   DMA+12,AL           ; SET THE FIRST/LA5T F/F
                        <1>       IODELAY                         ; WAIT FOR I/O
00004B99 EB00           <2>  jmp short $+2
00004B9B EB00           <2>  jmp short $+2
00004B9D E60B           <1>       OUT   DMA+11,AL           ; OUTPUT THE MODE BYTE
00004B9F 89D0           <1>       mov   eax, edx            ; Buffer address
00004BA1 E604           <1>       OUT   DMA+4,AL            ; OUTPUT LOW ADDRESS
                        <1>       IODELAY                         ; WAIT FOR I/O
00004BA3 EB00           <2>  jmp short $+2
00004BA5 EB00           <2>  jmp short $+2
00004BA7 88E0           <1>       MOV   AL,AH
00004BA9 E604           <1>       OUT   DMA+4,AL            ; OUTPUT HIGH ADDRESS
00004BAB C1E810         <1>       shr   eax, 16
                        <1>       IODELAY                         ; I/O WAIT STATE
00004BAE EB00           <2>  jmp short $+2
00004BB0 EB00           <2>  jmp short $+2
00004BB2 E681           <1>       OUT   081H,AL                   ; OUTPUT highest BITS TO PAGE REGISTER
                        <1>       IODELAY
00004BB4 EB00           <2>  jmp short $+2
00004BB6 EB00           <2>  jmp short $+2
00004BB8 6689C8         <1>       mov   ax, cx              ; Byte count - 1
00004BBB E605           <1>       OUT   DMA+5,AL            ; LOW BYTE OF COUNT
                        <1>       IODELAY                         ; WAIT FOR I/O
00004BBD EB00           <2>  jmp short $+2
00004BBF EB00           <2>  jmp short $+2
00004BC1 88E0           <1>       MOV   AL, AH
00004BC3 E605           <1>       OUT   DMA+5,AL            ; HIGH BYTE OF COUNT
                        <1>       IODELAY
00004BC5 EB00           <2>  jmp short $+2
00004BC7 EB00           <2>  jmp short $+2
00004BC9 FB             <1>       STI                       ; RE-ENABLE INTERRUPTS
00004BCA B002           <1>       MOV   AL, 2               ; MODE FOR 8237
00004BCC E60A           <1>       OUT   DMA+10, AL          ; INITIALIZE THE DISKETTE CHANNEL
                        <1>
00004BCE F8             <1>       clc   ; 04/02/2016
00004BCF C3             <1>       retn
                        <1>
                        <1> dma_bnd_err_stc:
00004BD0 F9             <1>       stc
                        <1> dma_bnd_err:
00004BD1 C605[18820100]09 <1>     MOV   byte [DSKETTE_STATUS], DMA_BOUNDARY ; SET ERROR
00004BD8 C3             <1>       RETn                      ; CY SET BY ABOVE IF ERROR
                        <1>
                        <1> ;; 16/12/2014
                        <1> ;;    CLI                       ; DISABLE INTERRUPTS DURING DMA SET-UP
                        <1> ;;    OUT   DMA+12,AL           ; SET THE FIRST/LA5T F/F
                        <1> ;;    ;JMP  $+2                 ; WAIT FOR I/O
                        <1> ;;    IODELAY
                        <1> ;;    OUT   DMA+11,AL           ; OUTPUT THE MODE BYTE
                        <1> ;;    ;SIODELAY
                        <1> ;;      ;CMP AL, 42H                  ; DMA VERIFY COMMAND
```

```
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 00004BD9 8B5504           <1>       mov    edx, [ebp+4]       ; Buffer address
1848 00004BDC F7C20000F0FF     <1>       test   edx, 0FFF00000h          ; 16 MB limit
1849 00004BE2 75EC             <1>       jnz    short dma_bnd_err_stc
1850                            <1>       ;
1851 00004BE4 6652             <1>       push   dx                 ; *
1852 00004BE6 B204             <1>       mov    DL, 4              ; SECTORS/TRACK VALUE IN PARM TABLE
1853 00004BE8 E8D0020000       <1>       call   GET_PARM           ; "
1854 00004BED 88E0             <1>       mov    al, ah             ; AL = SECTORS/TRACK VALUE
1855 00004BEF 28E4             <1>       sub    ah, ah             ; AX = SECTORS/TRACK VALUE
1856 00004BF1 66C1E002         <1>       shl    ax, 2              ; AX = SEC/TRK * 4 (OFFSET C,H,R,N)
1857 00004BF5 6648             <1>       dec    ax                 ; -1 FOR DMA VALUE
1858 00004BF7 6689C1           <1>       mov    cx, ax
1859 00004BFA 665A             <1>       pop    dx                 ; *
1860 00004BFC 6601CA           <1>       add    dx, cx             ; check for overflow
1861 00004BFF 72D0             <1>       jc     short dma_bnd_err
1862                            <1>       ;
1863 00004C01 6629CA           <1>       sub    dx, cx             ; Restore start address
1864                            <1>       ;
1865 00004C04 B04A             <1>       MOV    AL, 04AH           ; WILL WRITE TO THE DISKETTE
1866 00004C06 FA               <1>       CLI                       ; DISABLE INTERRUPTS DURING DMA SET-UP
1867 00004C07 E60C             <1>       OUT    DMA+12,AL          ; SET THE FIRST/LA5T F/F
1868                            <1>       IODELAY                         ; WAIT FOR I/O
1868 00004C09 EB00             <2>  jmp short $+2
1868 00004C0B EB00             <2>  jmp short $+2
1869 00004C0D E60B             <1>       OUT    DMA+11,AL          ; OUTPUT THE MODE BYTE
1870 00004C0F 89D0             <1>       mov    eax, edx           ; Buffer address
1871 00004C11 E604             <1>       OUT    DMA+4,AL           ; OUTPUT LOW ADDRESS
1872                            <1>       IODELAY                         ; WAIT FOR I/O
1872 00004C13 EB00             <2>  jmp short $+2
```

```
1872 00004C15 EB00              <2>  jmp short $+2
1873 00004C17 88E0              <1>       MOV    AL,AH
1874 00004C19 E604              <1>       OUT    DMA+4,AL           ; OUTPUT HIGH ADDRESS
1875 00004C1B C1E810            <1>       shr    eax, 16
1876                            <1>       IODELAY                         ; I/O WAIT STATE
1876 00004C1E EB00              <2>  jmp short $+2
1876 00004C20 EB00              <2>  jmp short $+2
1877 00004C22 E681              <1>       OUT    081H,AL                  ; OUTPUT highest BITS TO PAGE REGISTER
1878                            <1>       IODELAY
1878 00004C24 EB00              <2>  jmp short $+2
1878 00004C26 EB00              <2>  jmp short $+2
1879 00004C28 6689C8            <1>       mov    ax, cx             ; Byte count - 1
1880 00004C2B E605              <1>       OUT    DMA+5,AL           ; LOW BYTE OF COUNT
1881                            <1>       IODELAY                         ; WAIT FOR I/O
1881 00004C2D EB00              <2>  jmp short $+2
1881 00004C2F EB00              <2>  jmp short $+2
1882 00004C31 88E0              <1>       MOV    AL, AH
1883 00004C33 E605              <1>       OUT    DMA+5,AL           ; HIGH BYTE OF COUNT
1884                            <1>       IODELAY
1884 00004C35 EB00              <2>  jmp short $+2
1884 00004C37 EB00              <2>  jmp short $+2
1885 00004C39 FB                <1>       STI                       ; RE-ENABLE INTERRUPTS
1886 00004C3A B002              <1>       MOV    AL, 2              ; MODE FOR 8237
1887 00004C3C E60A              <1>       OUT    DMA+10, AL         ; INITIALIZE THE DISKETTE CHANNEL
1888 00004C3E 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>
1952                            <1> ;--------------------------------------------------------------------------------
1953                            <1> ; NEC_INIT
1954                            <1> ;     THIS ROUTINE SEEKS TO THE REQUESTED TRACK AND INITIALIZES
1955                            <1> ;     THE NEC FOR THE READ/WRITE/VERIFY/FORMAT OPERATION.
1956                            <1> ;
1957                            <1> ; ON ENTRY:  AH = NEC COMMAND TO BE PERFORMED
1958                            <1> ;
1959                            <1> ; ON EXIT:   @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
1960                            <1> ;--------------------------------------------------------------------------------
1961                            <1> NEC_INIT:
1962 00004C3F 6650              <1>       PUSH   AX                 ; SAVE NEC COMMAND
1963 00004C41 E8BC020000        <1>       CALL   MOTOR_ON           ; TURN MOTOR ON FOR SPECIFIC DRIVE
1964                            <1>
1965                            <1> ;-----      DO THE SEEK OPERATION
1966                            <1>
1967 00004C46 8A6D01            <1>       MOV    CH,[eBP+1]         ; CH = TRACK #
1968 00004C49 E8AF030000        <1>       CALL   SEEK               ; MOVE TO CORRECT TRACK
```

```
1969 00004C4E 6658              <1>       POP    AX                  ; RECOVER COMMAND
1970 00004C50 721E              <1>       JC     short ER_1          ; ERROR ON SEEK
1971 00004C52 BB[704C0000]      <1>       MOV    eBX, ER_1           ; LOAD ERROR ADDRESS
1972 00004C57 53                <1>       PUSH   eBX                 ; PUSH NEC_OUT ERROR RETURN
1973                            <1>
1974                            <1> ;-----      SEND OUT THE PARAMETERS TO THE CONTROLLER
1975                            <1>
1976 00004C58 E866030000        <1>       CALL   NEC_OUTPUT          ; OUTPUT THE OPERATION COMMAND
1977 00004C5D 6689F0            <1>       MOV    AX,SI               ; AH = HEAD #
1978 00004C60 89FB              <1>       MOV    eBX,eDI                   ; BL = DRIVE #
1979 00004C62 C0E402            <1>       SAL    AH,2                ; MOVE IT TO BIT 2
1980 00004C65 80E404            <1>       AND    AH,00000100B        ; ISOLATE THAT BIT
1981 00004C68 08DC              <1>       OR     AH,BL               ; OR IN THE DRIVE NUMBER
1982 00004C6A E854030000        <1>       CALL   NEC_OUTPUT          ; FALL THRU CY SET IF ERROR
1983 00004C6F 5B                <1>       POP    eBX                 ; THROW AWAY ERROR RETURN
1984                            <1> ER_1:
1985 00004C70 C3                <1>       RETn
1986                            <1>
1987                            <1> ;--------------------------------------------------------------------------------
1988                            <1> ; RWV_COM
1989                            <1> ;     THIS ROUTINE SENDS PARAMETERS TO THE NEC SPECIFIC TO THE
1990                            <1> ;     READ/WRITE/VERIFY OPERATIONS.
1991                            <1> ;
1992                            <1> ; ON ENTRY:  CS:BX = ADDRESS OF MEDIA/DRIVE PARAMETER TABLE
1993                            <1> ; ON EXIT:   @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
1994                            <1> ;--------------------------------------------------------------------------------
1995                            <1> RWV_COM:
1996 00004C71 B8[BC4C0000]      <1>       MOV    eAX, ER_2           ; LOAD ERROR ADDRESS
1997 00004C76 50                <1>       PUSH   eAX                 ; PUSH NEC_OUT ERROR RETURN
1998 00004C77 8A6501            <1>       MOV    AH,[eBP+1]          ; OUTPUT TRACK #
1999 00004C7A E844030000        <1>       CALL   NEC_OUTPUT
2000 00004C7F 6689F0            <1>       MOV    AX,SI               ; OUTPUT HEAD #
2001 00004C82 E83C030000        <1>       CALL   NEC_OUTPUT
2002 00004C87 8A6500            <1>         MOV     AH,[eBP]                ; OUTPUT SECTOR #
2003 00004C8A E834030000        <1>       CALL   NEC_OUTPUT
2004 00004C8F B203              <1>       MOV    DL,3                ; BYTES/SECTOR PARAMETER FROM BLOCK
2005 00004C91 E827020000        <1>       CALL   GET_PARM            ; ... TO THE NEC
2006 00004C96 E828030000        <1>       CALL   NEC_OUTPUT          ; OUTPUT TO CONTROLLER
2007 00004C9B B204              <1>       MOV    DL,4                ; EOT PARAMETER FROM BLOCK
2008 00004C9D E81B020000        <1>       CALL   GET_PARM            ; ... TO THE NEC
2009 00004CA2 E81C030000        <1>       CALL   NEC_OUTPUT          ; OUTPUT TO CONTROLLER
2010 00004CA7 8A6305            <1>         MOV     AH, [eBX+MD.GAP]        ; GET GAP LENGTH
2011                            <1> _R15:
2012 00004CAA E814030000        <1>       CALL   NEC_OUTPUT
2013 00004CAF B206              <1>       MOV    DL,6                ; DTL PARAMETER PROM BLOCK
2014 00004CB1 E807020000        <1>       CALL   GET_PARM            ;  TO THE NEC
2015 00004CB6 E808030000        <1>       CALL   NEC_OUTPUT          ; OUTPUT TO CONTROLLER
2016 00004CBB 58                <1>       POP    eAX                 ; THROW AWAY ERROR EXIT
2017                            <1> ER_2:
2018 00004CBC C3                <1>       RETn
2019                            <1>
2020                            <1> ;--------------------------------------------------------------------------------
2021                            <1> ; NEC_TERM
2022                            <1> ;     THIS ROUTINE WAITS FOR THE OPERATION THEN ACCEPTS THE STATUS
2023                            <1> ;     FROM THE NEC FOR THE READ/WRITE/VERIFY/FORWAT OPERATION.
2024                            <1> ;
2025                            <1> ; ON EXIT:   @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
2026                            <1> ;--------------------------------------------------------------------------------
2027                            <1> NEC_TERM:
2028                            <1>
2029                            <1> ;-----      LET THE OPERATION HAPPEN
2030                            <1>
2031 00004CBD 56                <1>       PUSH   eSI                 ; SAVE HEAD #, # OF SECTORS
2032 00004CBE E80D040000        <1>       CALL   WAIT_INT            ; WAIT FOR THE INTERRUPT
2033 00004CC3 9C                <1>       PUSHF
2034 00004CC4 E837040000        <1>       CALL   RESULTS                   ; GET THE NEC STATUS
2035 00004CC9 724B              <1>       JC     short SET_END_POP
2036 00004CCB 9D                <1>       POPF
2037 00004CCC 723E              <1>       JC     short SET_END       ; LOOK FOR ERROR
2038                            <1>
2039                            <1> ;-----      CHECK THE RESULTS RETURNED BY THE CONTROLLER
2040                            <1>
2041 00004CCE FC                <1>       CLD                        ; SET THE CORRECT DIRECTION
2042 00004CCF BE[19820100]      <1>       MOV    eSI, NEC_STATUS           ; POINT TO STATUS FIELD
2043 00004CD4 AC                <1>       lodsb                      ; GET ST0
2044 00004CD5 24C0              <1>       AND    AL,11000000B        ; TEST FOR NORMAL TERMINATION
2045 00004CD7 7433              <1>       JZ     short SET_END
2046 00004CD9 3C40              <1>       CMP    AL,01000000B        ; TEST FOR ABNORMAL TERMINATION
2047 00004CDB 7527              <1>       JNZ    short J18           ; NOT ABNORMAL, BAD NEC
2048                            <1>
2049                            <1> ;-----      ABNORMAL TERMINATION, FIND OUT WHY
2050                            <1>
2051 00004CDD AC                <1>       lodsb                      ; GET ST1
2052 00004CDE D0E0              <1>       SAL    AL,1                ; TEST FOR EDT FOUND
2053 00004CE0 B404              <1>       MOV    AH,RECORD_NOT_FND
2054 00004CE2 7222              <1>       JC     short J19
2055 00004CE4 C0E002            <1>       SAL    AL,2
2056 00004CE7 B410              <1>       MOV    AH,BAD_CRC
2057 00004CE9 721B              <1>       JC     short J19
2058 00004CEB D0E0              <1>       SAL    AL,1                ; TEST FOR DMA OVERRUN
2059 00004CED B408              <1>       MOV    AH,BAD_DMA
2060 00004CEF 7215              <1>       JC     short J19
2061 00004CF1 C0E002            <1>       SAL    AL,2                ; TEST FOR RECORD NOT FOUND
2062 00004CF4 B404              <1>       MOV    AH,RECORD_NOT_FND
2063 00004CF6 720E              <1>       JC     short J19
2064 00004CF8 D0E0              <1>       SAL    AL,1
2065 00004CFA B403              <1>       MOV    AH,WRITE_PROTECT    ; TEST FOR WRITE_PROTECT
2066 00004CFC 7208              <1>       JC     short J19
2067 00004CFE D0E0              <1>       SAL    AL,1                ; TEST MISSING ADDRESS MARK
2068 00004D00 B402              <1>       MOV    AH,BAD_ADDR_MARK
2069 00004D02 7202              <1>       JC     short J19
2070                            <1>
2071                            <1> ;-----      NEC MUST HAVE FAILED
2072                            <1> J18:
2073 00004D04 B420              <1>       MOV    AH,BAD_NEC
```

```
2074                                    <1> J19:
2075 00004D06 0825[18820100]            <1>         OR      [DSKETTE_STATUS], AH
2076                                    <1> SET_END:
2077 00004D0C 803D[18820100]01          <1>         CMP     byte [DSKETTE_STATUS], 1 ; SET ERROR CONDITION
2078 00004D13 F5                        <1>         CMC
2079 00004D14 5E                        <1>         POP     eSI
2080 00004D15 C3                        <1>         RETn                        ; RESTORE HEAD #, # OF SECTORS
2081                                    <1>
2082                                    <1> SET_END_POP:
2083 00004D16 9D                        <1>         POPF
2084 00004D17 EBF3                      <1>         JMP     SHORT SET_END
2085                                    <1>
2086                                    <1> ;--------------------------------------------------------------------------
2087                                    <1> ; DSTATE:   ESTABLISH STATE UPON SUCCESSFUL OPERATION.
2088                                    <1> ;--------------------------------------------------------------------------
2089                                    <1> DSTATE:
2090 00004D19 803D[18820100]00          <1>         CMP     byte [DSKETTE_STATUS],0    ; CHECK FOR ERROR
2091 00004D20 753E                      <1>         JNZ     short SETBAC               ; IF ERROR JUMP
2092 00004D22 808F[25820100]10          <1>         OR      byte [DSK_STATE+eDI],MED_DET ; NO ERROR, MARK MEDIA AS DETERMINED
2093 00004D29 F687[25820100]04          <1>         TEST    byte [DSK_STATE+eDI],DRV_DET ; DRIVE DETERMINED ?
2094 00004D30 752E                      <1>         JNZ     short SETBAC        ; IF DETERMINED NO TRY TO DETERMINE
2095 00004D32 8A87[25820100]            <1>         MOV     AL,[DSK_STATE+eDI]  ; LOAD STATE
2096 00004D38 24C0                      <1>         AND     AL,RATE_MSK         ; KEEP ONLY RATE
2097 00004D3A 3C80                      <1>         CMP     AL,RATE_250         ; RATE 250 ?
2098 00004D3C 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 00004D3E E871010000                <1>         CALL    CMOS_TYPE           ; RETURN DRIVE TYPE IN (AL)
2103                                    <1>         ;;20/02/2015
2104                                    <1>         ;;JC    short M_12          ; CMOS BAD
2105 00004D43 7414                      <1>         jz      short M_12 ;; 20/02/2015
2106 00004D45 3C04                      <1>         CMP     AL, 4               ; 1.44MB DRIVE ?
2107 00004D47 7410                      <1>         JE      short M_12          ; YES
2108                                    <1> M_720:
2109 00004D49 80A7[25820100]FD          <1>         AND     byte [DSK_STATE+eDI], ~FMT_CAPA ; TURN OFF FORMAT CAPABILITY
2110 00004D50 808F[25820100]04          <1>         OR      byte [DSK_STATE+eDI],DRV_DET  ; MARK DRIVE DETERMINED
2111 00004D57 EB07                      <1>         JMP     SHORT SETBAC        ; BACK
2112                                    <1> M_12:
2113 00004D59 808F[25820100]06          <1>         OR      byte [DSK_STATE+eDI],DRV_DET+FMT_CAPA
2114                                    <1>                                     ; TURN ON DETERMINED & FMT CAPA
2115                                    <1> SETBAC:
2116 00004D60 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 00004D61 803D[18820100]00          <1>         CMP     byte [DSKETTE_STATUS],0    ; GET STATUS OF OPERATION
2127 00004D68 7445                      <1>         JZ      short NO_RETRY             ; SUCCESSFUL OPERATION
2128 00004D6A 803D[18820100]80          <1>         CMP     byte [DSKETTE_STATUS],TIME_OUT ; IF TIME OUT NO RETRY
2129 00004D71 743C                      <1>         JZ      short NO_RETRY
2130 00004D73 8AA7[25820100]            <1>         MOV     AH,[DSK_STATE+eDI]  ; GET MEDIA STATE OF DRIVE
2131 00004D79 F6C410                    <1>         TEST    AH,MED_DET          ; ESTABLISHED/DETERMINED ?
2132 00004D7C 7531                      <1>         JNZ     short NO_RETRY            ; IF ESTABLISHED STATE THEN TRUE ERROR
2133 00004D7E 80E4C0                    <1>         AND     AH,RATE_MSK         ; ISOLATE RATE
2134 00004D81 8A2D[20820100]            <1>         MOV     CH,[LASTRATE]       ; GET START OPERATION STATE
2135 00004D87 C0C504                    <1>         ROL     CH,4                ; TO CORRESPONDING BITS
2136 00004D8A 80E5C0                    <1>         AND     CH,RATE_MSK         ; ISOLATE RATE BITS
2137 00004D8D 38E5                      <1>         CMP     CH,AH               ; ALL RATES TRIED
2138 00004D8F 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 00004D91 80FC01                    <1>         CMP     AH,RATE_500+1       ; SET CY FOR RATE 500
2146 00004D94 D0DC                      <1>         RCR     AH,1                ; TO NEXT STATE
2147 00004D96 80E4C0                    <1>         AND     AH,RATE_MSK         ; KEEP ONLY RATE BITS
2148 00004D99 80A7[25820100]1F          <1>         AND     byte [DSK_STATE+eDI], ~(RATE_MSK+DBL_STEP)
2149                                    <1>                                     ; RATE, DBL STEP OFF
2150 00004DA0 08A7[25820100]            <1>         OR      [DSK_STATE+eDI],AH  ; TURN ON NEW RATE
2151 00004DA6 C605[18820100]00          <1>         MOV     byte [DSKETTE_STATUS],0    ; RESET STATUS FOR RETRY
2152 00004DAD F9                        <1>         STC                         ; SET CARRY FOR RETRY
2153 00004DAE C3                        <1>         RETn                        ; RETRY RETURN
2154                                    <1>
2155                                    <1> NO_RETRY:
2156 00004DAF F8                        <1>         CLC                         ; CLEAR CARRY NO RETRY
2157 00004DB0 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
2162                                    <1> ;     ACTUALLY TRANSFERRED TO/FROM THE DISKETTE.
2163                                    <1> ;
2164                                    <1> ; ON ENTRY:  [BP+1] = TRACK
2165                                    <1> ;            SI-HI  = HEAD
2166                                    <1> ;            [BP]   = START SECTOR
2167                                    <1> ;
2168                                    <1> ; ON EXIT:   AL = NUMBER ACTUALLY TRANSFERRED
2169                                    <1> ;--------------------------------------------------------------------------
2170                                    <1> NUM_TRANS:
2171 00004DB1 30C0                      <1>         XOR     AL,AL               ; CLEAR FOR ERROR
2172 00004DB3 803D[18820100]00          <1>         CMP     byte [DSKETTE_STATUS],0    ; CHECK FOR ERROR
2173 00004DBA 752C                      <1>         JNZ     NT_OUT              ; IF ERROR 0 TRANSFERRED
2174 00004DBC B204                      <1>         MOV     DL,4                ; SECTORS/TRACK OFFSET TO DL
2175 00004DBE E8FA000000                <1>         CALL    GET_PARM            ; AH = SECTORS/TRACK
2176 00004DC3 8A1D[1E820100]            <1>         MOV     BL, [NEC_STATUS+5]  ; GET ENDING SECTOR
2177 00004DC9 6689F1                    <1>         MOV     CX,SI               ; CH = HEAD # STARTED
2178 00004DCC 3A2D[1D820100]            <1>         CMP     CH, [NEC_STATUS+4]  ; GET HEAD ENDED UP ON
```

```
2179 00004DD2 750D              <1>       JNZ    DIF_HD              ; IF ON SAME HEAD, THEN NO ADJUST
2180 00004DD4 8A2D[1C820100]    <1>       MOV    CH, [NEC_STATUS+3]  ; GET TRACK ENDED UP ON
2181 00004DDA 3A6D01            <1>       CMP    CH,[eBP+1]          ; IS IT ASKED FOR TRACK
2182 00004DDD 7404              <1>       JZ     short SAME_TRK            ; IF SAME TRACK NO INCREASE
2183 00004DDF 00E3              <1>       ADD    BL,AH               ; ADD SECTORS/TRACK
2184                            <1> DIF_HD:
2185 00004DE1 00E3              <1>       ADD    BL,AH               ; ADD SECTORS/TRACK
2186                            <1> SAME_TRK:
2187 00004DE3 2A5D00            <1>       SUB    BL,[eBP]            ; SUBTRACT START FROM END
2188 00004DE6 88D8              <1>       MOV    AL,BL               ; TO AL
2189                            <1> NT_OUT:
2190 00004DE8 C3                <1>       RETn
2191                            <1>
2192                            <1> ;--------------------------------------------------------------------------
2193                            <1> ; SETUP_END
2194                            <1> ;     RESTORES @MOTOR_COUNT TO PARAMETER PROVIDED IN TABLE
2195                            <1> ;     AND LOADS @DSKETTE_STATUS TO AH, AND SETS CY.
2196                            <1> ;
2197                            <1> ; ON EXIT:
2198                            <1> ;     AH, @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
2199                            <1> ;--------------------------------------------------------------------------
2200                            <1> SETUP_END:
2201 00004DE9 B202              <1>       MOV    DL,2                ; GET THE MOTOR WAIT PARAMETER
2202 00004DEB 6650              <1>       PUSH   AX                  ; SAVE NUMBER TRANSFERRED
2203 00004DED E8CB000000        <1>       CALL   GET_PARM
2204 00004DF2 8825[17820100]    <1>       MOV    [MOTOR_COUNT],AH    ; STORE UPON RETURN
2205 00004DF8 6658              <1>       POP    AX                  ; RESTORE NUMBER TRANSFERRED
2206 00004DFA 8A25[18820100]    <1>       MOV    AH, [DSKETTE_STATUS]      ; GET STATUS OF OPERATION
2207 00004E00 08E4              <1>       OR     AH,AH               ; CHECK FOR ERROR
2208 00004E02 7402              <1>       JZ     short NUN_ERR       ; NO ERROR
2209 00004E04 30C0              <1>       XOR    AL,AL               ; CLEAR NUMBER RETURNED
2210                            <1> NUN_ERR:
2211 00004E06 80FC01            <1>       CMP    AH,1                ; SET THE CARRY FLAG TO INDICATE
2212 00004E09 F5                <1>       CMC                        ; SUCCESS OR FAILURE
2213 00004E0A C3                <1>       RETn
2214                            <1>
2215                            <1> ;--------------------------------------------------------------------------
2216                            <1> ; SETUP_DBL
2217                            <1> ;     CHECK DOUBLE STEP.
2218                            <1> ;
2219                            <1> ; ON ENTRY : DI = DRIVE
2220                            <1> ;
2221                            <1> ; ON EXIT :  CY = 1 MEANS ERROR
2222                            <1> ;--------------------------------------------------------------------------
2223                            <1> SETUP_DBL:
2224 00004E0B 8AA7[25820100]    <1>       MOV    AH, [DSK_STATE+eDI] ; ACCESS STATE
2225 00004E11 F6C410            <1>       TEST   AH,MED_DET          ; ESTABLISHED STATE ?
2226 00004E14 757E              <1>       JNZ    short NO_DBL              ; IF ESTABLISHED THEN DOUBLE DONE
2227                            <1>
2228                            <1> ;-----      CHECK FOR TRACK 0 TO SPEED UP ACKNOWLEDGE OF UNFORMATTED DISKETTE
2229                            <1>
2230 00004E16 C605[15820100]00  <1>       MOV    byte [SEEK_STATUS],0      ; SET RECALIBRATE REQUIRED ON ALL DRIVES
2231 00004E1D E8E0000000        <1>       CALL   MOTOR_ON            ; ENSURE MOTOR STAY ON
2232 00004E22 B500              <1>       MOV    CH,0                ; LOAD TRACK 0
2233 00004E24 E8D4010000        <1>       CALL   SEEK                ; SEEK TO TRACK 0
2234 00004E29 E868000000        <1>       CALL   READ_ID                   ; READ ID FUNCTION
2235 00004E2E 7249              <1>       JC     short SD_ERR        ; IF ERROR NO TRACK 0
2236                            <1>
2237                            <1> ;-----      INITIALIZE START AND MAX TRACKS (TIMES 2 FOR BOTH HEADS)
2238                            <1>
2239 00004E30 66B95004          <1>       MOV    CX,0450H            ; START, MAX TRACKS
2240 00004E34 F687[25820100]01  <1>       TEST   byte [DSK_STATE+eDI],TRK_CAPA ; TEST FOR 80 TRACK CAPABILITY
2241 00004E3B 7402              <1>       JZ     short CNT_OK        ; IF NOT COUNT IS SETUP
2242 00004E3D B1A0              <1>       MOV    CL,0A0H                  ; MAXIMUM TRACK 1.2 MB
2243                            <1>
2244                            <1> ;     ATTEMPT READ ID OF ALL TRACKS, ALL HEADS UNTIL SUCCESS; UPON SUCCESS,
2245                            <1> ;     MUST SEE IF ASKED FOR TRACK IN SINGLE STEP MODE = TRACK ID READ; IF NOT
2246                            <1> ;     THEN SET DOUBLE STEP ON.
2247                            <1>
2248                            <1> CNT_OK:
2249 00004E3F C605[17820100]FF  <1>         MOV     byte [MOTOR_COUNT], 0FFH ; ENSURE MOTOR STAYS ON FOR OPERATION
2250 00004E46 6651              <1>       PUSH   CX                  ; SAVE TRACK, COUNT
2251 00004E48 C605[18820100]00  <1>       MOV    byte [DSKETTE_STATUS],0   ; CLEAR STATUS, EXPECT ERRORS
2252 00004E4F 6631C0            <1>       XOR    AX,AX               ; CLEAR AX
2253 00004E52 D0ED              <1>       SHR    CH,1                ; HALVE TRACK, CY = HEAD
2254 00004E54 C0D003            <1>       RCL    AL,3                ; AX = HEAD IN CORRECT BIT
2255 00004E57 6650              <1>       PUSH   AX                  ; SAVE HEAD
2256 00004E59 E89F010000        <1>       CALL   SEEK                ; SEEK TO TRACK
2257 00004E5E 6658              <1>       POP    AX                  ; RESTORE HEAD
2258 00004E60 6609C7            <1>       OR     DI,AX               ; DI = HEAD OR'ED DRIVE
2259 00004E63 E82E000000        <1>       CALL   READ_ID                   ; READ ID HEAD 0
2260 00004E68 9C                <1>       PUSHF                      ; SAVE RETURN FROM READ_ID
2261 00004E69 6681E7FB00        <1>       AND    DI,11111011B        ; TURN OFF HEAD 1 BIT
2262 00004E6E 9D                <1>       POPF                       ; RESTORE ERROR RETURN
2263 00004E6F 6659              <1>       POP    CX                  ; RESTORE COUNT
2264 00004E71 7308              <1>       JNC    short DO_CHK        ; IF OK, ASKED = RETURNED TRACK ?
2265 00004E73 FEC5              <1>       INC    CH                  ; INC FOR NEXT TRACK
2266 00004E75 38CD              <1>       CMP    CH,CL               ; REACHED MAXIMUM YET
2267 00004E77 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 00004E79 F9                <1>       STC                        ; SET CARRY FOR ERROR
2273 00004E7A C3                <1>       RETn                       ; SETUP_DBL ERROR EXIT
2274                            <1>
2275                            <1> DO_CHK:
2276 00004E7B 8A0D[1C820100]    <1>       MOV    CL, [NEC_STATUS+3]  ; LOAD RETURNED TRACK
2277 00004E81 888F[29820100]    <1>       MOV    [DSK_TRK+eDI], CL   ; STORE TRACK NUMBER
2278 00004E87 D0ED              <1>       SHR    CH,1                ; HALVE TRACK
2279 00004E89 38CD              <1>       CMP    CH,CL               ; IS IT THE SAME AS ASKED FOR TRACK
2280 00004E8B 7407              <1>       JZ     short NO_DBL        ; IF SAME THEN NO DOUBLE STEP
2281 00004E8D 808F[25820100]20  <1>       OR     byte [DSK_STATE+eDI],DBL_STEP ; TURN ON DOUBLE STEP REQUIRED
2282                            <1> NO_DBL:
2283 00004E94 F8                <1>       CLC                        ; CLEAR ERROR FLAG
```

```
2284 00004E95 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 00004E96 B8[B34E0000]      <1>       MOV    eAX, ER_3          ; MOVE NEC OUTPUT ERROR ADDRESS
2297 00004E9B 50                <1>       PUSH   eAX
2298 00004E9C B44A              <1>       MOV    AH,4AH             ; READ ID COMMAND
2299 00004E9E E820010000        <1>       CALL   NEC_OUTPUT         ; TO CONTROLLER
2300 00004EA3 6689F8            <1>       MOV    AX,DI              ; DRIVE # TO AH, HEAD 0
2301 00004EA6 88C4              <1>       MOV    AH,AL
2302 00004EA8 E816010000        <1>       CALL   NEC_OUTPUT         ; TO CONTROLLER
2303 00004EAD E80BFEFFFF        <1>       CALL   NEC_TERM           ; WAIT FOR OPERATION, GET STATUS
2304 00004EB2 58                <1>       POP    eAX                ; THROW AWAY ERROR ADDRESS
2305                            <1> ER_3:
2306 00004EB3 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 00004EB4 8A87[EE6D0000]    <1> mov   al, [eDI+fd0_type]
2319 00004EBA 20C0              <1> and   al, al ; 18/12/2014
2320 00004EBC 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 00004EBD 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 00004EBE 87D3              <1>       XCHG   eDX,eBX                   ; BL = INDEX
2359                            <1>       ;SUB   BH,BH               ; BX = INDEX
2360 00004EC0 81E3FF000000      <1>       and    ebx, 0FFh
2361                            <1>       ;LDS   SI, [DISK_POINTER]  ; POINT TO BLOCK
2362                            <1>       ;
2363                            <1>       ; 17/12/2014
2364 00004EC6 66A1[DD6D0000]    <1>       mov    ax, [cfd] ; current (AL) and previous fd (AH)
2365 00004ECC 38E0              <1>       cmp    al, ah
2366 00004ECE 7425              <1>       je     short gpndc
2367 00004ED0 A2[DE6D0000]      <1>       mov    [pfd], al ; current drive -> previous drive
2368 00004ED5 53                <1>       push   ebx ; 08/02/2015
2369 00004ED6 88C3              <1>       mov    bl, al
2370                            <1>       ; 11/12/2014
2371 00004ED8 8A83[EE6D0000]    <1>       mov    al, [eBX+fd0_type]  ; Drive type (0,1,2,3,4)
2372                            <1>       ; 18/12/2014
2373 00004EDE 20C0              <1>       and    al, al
2374 00004EE0 7507              <1>       jnz    short gpdtc
2375 00004EE2 BB[C76D0000]      <1>       mov    ebx, MD_TBL6        ; 1.44 MB param. tbl. (default)
2376 00004EE7 EB05              <1>         jmp     short gpdpu
2377                            <1> gpdtc:
2378 00004EE9 E817F9FFFF        <1>       call   DR_TYPE_CHECK
2379                            <1>       ; cf = 1 -> eBX points to 1.44MB fd parameter table (default)
2380                            <1> gpdpu:
2381 00004EEE 891D[646D0000]    <1>       mov    [DISK_POINTER], ebx
2382 00004EF4 5B                <1>       pop    ebx
2383                            <1> gpndc:
2384 00004EF5 8B35[646D0000]    <1>       mov    esi, [DISK_POINTER] ; 08/02/2015, si -> esi
2385 00004EFB 8A241E            <1>       MOV    AH, [eSI+eBX]       ; GET THE WORD
2386 00004EFE 87D3              <1>       XCHG   eDX,eBX                   ; RESTORE BX
2387 00004F00 5E                <1>       POP    eSI
2388                            <1>       ;POP   DS
```

```
2389 00004F01 C3                <1>       RETn
2390                            <1>
2391                            <1> ;--------------------------------------------------------------------------------
2392                            <1> ; MOTOR_ON
2393                            <1> ;     TURN MOTOR ON AND WAIT FOR MOTOR START UP TIME. THE @MOTOR_COUNT
2394                            <1> ;     IS REPLACED WITH A SUFFICIENTLY HIGH NUMBER (0FFH) TO ENSURE
2395                            <1> ;     THAT THE MOTOR DOES NOT GO OFF DURING THE OPERATION. IF THE
2396                            <1> ;     MOTOR NEEDED TO BE TURNED ON, THE MULTI-TASKING HOOK FUNCTION
2397                            <1> ;     (AX=90FDH, INT 15) IS CALLED TELLING THE OPERATING SYSTEM
2398                            <1> ;     THAT THE BIOS IS ABOUT TO WAIT FOR MOTOR START UP. IF THIS
2399                            <1> ;     FUNCTION RETURNS WITH CY = 1, IT MEANS THAT THE MINIMUM WAIT
2400                            <1> ;     HAS BEEN COMPLETED. AT THIS POINT A CHECK IS MADE TO ENSURE
2401                            <1> ;     THAT THE MOTOR WASN'T TURNED OFF BY THE TIMER. IF THE HOOK DID
2402                            <1> ;     NOT WAIT, THE WAIT FUNCTION (AH=086H) IS CALLED TO WAIT THE
2403                            <1> ;     PRESCRIBED AMOUNT OF TIME. IF THE CARRY FLAG IS SET ON RETURN,
2404                            <1> ;     IT MEANS THAT THE FUNCTION IS IN USE AND DID NOT PERFORM THE
2405                            <1> ;     WAIT. A TIMER 1 WAIT LOOP WILL THEN DO THE WAIT.
2406                            <1> ;
2407                            <1> ; ON ENTRY:  DI = DRIVE #
2408                            <1> ; ON EXIT:   AX,CX,DX DESTROYED
2409                            <1> ;--------------------------------------------------------------------------------
2410                            <1> MOTOR_ON:
2411 00004F02 53               <1>       PUSH   eBX                ; SAVE REG.
2412 00004F03 E82A000000       <1>       CALL   TURN_ON                  ; TURN ON MOTOR
2413 00004F08 7226             <1>       JC     short MOT_IS_ON          ; IF CY=1 NO WAIT
2414 00004F0A E89BF9FFFF       <1>       CALL   XLAT_OLD           ; TRANSLATE STATE TO COMPATIBLE MODE
2415 00004F0F E865F9FFFF       <1>       CALL   XLAT_NEW           ; TRANSLATE STATE TO PRESENT ARCH,
2416                            <1>       ;CALL  TURN_ON            ; CHECK AGAIN IF MOTOR ON
2417                            <1>       ;JC    MOT_IS_ON          ; IF NO WAIT MEANS IT IS ON
2418                            <1> M_WAIT:
2419 00004F14 B20A             <1>       MOV    DL,10              ; GET THE MOTOR WAIT PARAMETER
2420 00004F16 E8A2FFFFFF       <1>       CALL   GET_PARM
2421                            <1>       ;MOV   AL,AH              ; AL = MOTOR WAIT PARAMETER
2422                            <1>       ;XOR   AH,AH              ; AX = MOTOR WAIT PARAMETER
2423                            <1>       ;CMP   AL,8               ; SEE IF AT LEAST A SECOND IS SPECIFIED
2424 00004F1B 80FC08           <1>       cmp    ah, 8
2425                            <1>       ;JAE   short GP2          ; IF YES, CONTINUE
2426 00004F1E 7702             <1>       ja     short J13
2427                            <1>       ;MOV   AL,8               ; ONE SECOND WAIT FOR MOTOR START UP
2428 00004F20 B408             <1>       mov    ah, 8
2429                            <1>
2430                            <1> ;-----      AS CONTAINS NUMBER OF 1/8 SECONDS (125000 MICROSECONDS) TO WAIT
2431                            <1> GP2:
2432                            <1> ;-----      FOLLOWING LOOPS REQUIRED WHEN RTC WAIT FUNCTION IS ALREADY IN USE
2433                            <1> J13:                          ; WAIT FOR 1/8 SECOND PER (AL)
2434 00004F22 B95E200000       <1>       MOV    eCX,8286          ; COUNT FOR 1/8 SECOND AT 15.085737 US
2435 00004F27 E810D5FFFF       <1>       CALL   WAITF             ; GO TO FIXED WAIT ROUTINE
2436                            <1>       ;DEC   AL                 ; DECREMENT TIME VALUE
2437 00004F2C FECC             <1>       dec    ah
2438 00004F2E 75F2             <1>       JNZ    short J13          ; ARE WE DONE YET
2439                            <1> MOT_IS_ON:
2440 00004F30 5B               <1>       POP    eBX                ; RESTORE REG.
2441 00004F31 C3               <1>       RETn
2442                            <1>
2443                            <1> ;--------------------------------------------------------------------------------
2444                            <1> ; TURN_ON
2445                            <1> ;     TURN MOTOR ON AND RETURN WAIT STATE.
2446                            <1> ;
2447                            <1> ; ON ENTRY:  DI = DRIVE #
2448                            <1> ;
2449                            <1> ; ON EXIT:   CY = 0 MEANS WAIT REQUIRED
2450                            <1> ;            CY = 1 MEANS NO WAIT REQUIRED
2451                            <1> ;            AX,BX,CX,DX DESTROYED
2452                            <1> ;--------------------------------------------------------------------------------
2453                            <1> TURN_ON:
2454 00004F32 89FB             <1>       MOV    eBX,eDI                  ; BX = DRIVE #
2455 00004F34 88D9             <1>       MOV    CL,BL              ; CL = DRIVE #
2456 00004F36 C0C304           <1>       ROL    BL,4               ; BL = DRIVE SELECT
2457 00004F39 FA               <1>       CLI                       ; NO INTERRUPTS WHILE DETERMINING STATUS
2458 00004F3A C605[17820100]FF <1>       MOV    byte [MOTOR_COUNT],0FFH   ; ENSURE MOTOR STAYS ON FOR OPERATION
2459 00004F41 A0[16820100]     <1>       MOV    AL, [MOTOR_STATUS] ; GET DIGITAL OUTPUT REGISTER REFLECTION
2460 00004F46 2430             <1>       AND    AL,00110000B       ; KEEP ONLY DRIVE SELECT BITS
2461 00004F48 B401             <1>       MOV    AH,1               ; MASK FOR DETERMINING MOTOR BIT
2462 00004F4A D2E4             <1>       SHL    AH,CL              ; AH = MOTOR ON, A=00000001, B=00000010
2463                            <1>
2464                            <1> ;  AL = DRIVE SELECT FROM @MOTOR_STATUS
2465                            <1> ;  BL = DRIVE SELECT DESIRED
2466                            <1> ;  AH = MOTOR ON MASK DESIRED
2467                            <1>
2468 00004F4C 38D8             <1>       CMP    AL,BL              ; REQUESTED DRIVE ALREADY SELECTED ?
2469 00004F4E 7508             <1>       JNZ    short TURN_IT_ON   ; IF NOT SELECTED JUMP
2470 00004F50 8425[16820100]   <1>       TEST   AH, [MOTOR_STATUS] ; TEST MOTOR ON BIT
2471 00004F56 7535             <1>       JNZ    short NO_MOT_WAIT  ; JUMP IF MOTOR ON AND SELECTED
2472                            <1>
2473                            <1> TURN_IT_ON:
2474 00004F58 08DC             <1>       OR     AH,BL              ; AH = DRIVE SELECT AND MOTOR ON
2475 00004F5A 8A3D[16820100]   <1>       MOV    BH,[MOTOR_STATUS]  ; SAVE COPY OF @MOTOR_STATUS BEFORE
2476 00004F60 80E70F           <1>       AND    BH,00001111B       ; KEEP ONLY MOTOR BITS
2477 00004F63 8025[16820100]CF <1>       AND    byte [MOTOR_STATUS],11001111B ; CLEAR OUT DRIVE SELECT
2478 00004F6A 0825[16820100]   <1>       OR     [MOTOR_STATUS],AH  ; OR IN DRIVE SELECTED AND MOTOR ON
2479 00004F70 A0[16820100]     <1>       MOV    AL,[MOTOR_STATUS]  ; GET DIGITAL OUTPUT REGISTER REFLECTION
2480 00004F75 88C3             <1>       MOV    BL,AL              ; BL=@MOTOR_STATUS AFTER, BH=BEFORE
2481 00004F77 80E30F           <1>       AND    BL,00001111B       ; KEEP ONLY MOTOR BITS
2482 00004F7A FB               <1>       STI                       ; ENABLE INTERRUPTS AGAIN
2483 00004F7B 243F             <1>       AND    AL,00111111B       ; STRIP AWAY UNWANTED BITS
2484 00004F7D C0C004           <1>       ROL    AL,4               ; PUT BITS IN DESIRED POSITIONS
2485 00004F80 0C0C             <1>       OR     AL,00001100B       ; NO RESET, ENABLE DMA/INTERRUPT
2486 00004F82 66BAF203         <1>       MOV    DX,03F2H           ; SELECT DRIVE AND TURN ON MOTOR
2487 00004F86 EE               <1>       OUT    DX,AL
2488 00004F87 38FB             <1>       CMP    BL,BH              ; NEW MOTOR TURNED ON ?
2489                            <1>       ;JZ    short NO_MOT_WAIT  ; NO WAIT REQUIRED IF JUST SELECT
2490 00004F89 7403             <1>       je     short no_mot_w1 ; 27/02/2015
2491 00004F8B F8               <1>       CLC                       ; (re)SET CARRY MEANING WAIT
2492 00004F8C C3               <1>       RETn
2493                            <1>
```

```
                            <1> NO_MOT_WAIT:
00004F8D FB                 <1>         sti
                            <1> no_mot_w1: ; 27/02/2015
00004F8E F9                 <1>         STC                         ; SET NO WAIT REQUIRED
                            <1>         ;STI                        ; INTERRUPTS BACK ON
00004F8F C3                 <1>         RETn
                            <1>
                            <1> ;--------------------------------------------------------------------------
                            <1> ; HD_WAIT
                            <1> ;       WAIT FOR HEAD SETTLE TIME.
                            <1> ;
                            <1> ; ON ENTRY:  DI = DRIVE #
                            <1> ;
                            <1> ; ON EXIT:   AX,BX,CX,DX DESTROYED
                            <1> ;--------------------------------------------------------------------------
                            <1> HD_WAIT:
00004F90 B209               <1>         MOV     DL,9                ; GET HEAD SETTLE PARAMETER
00004F92 E826FFFFFF         <1>         CALL    GET_PARM
00004F97 08E4               <1>         or      ah, ah ; 17/12/2014
00004F99 7519               <1>         jnz     short DO_WAT
00004F9B F605[16820100]80   <1>           TEST    byte [MOTOR_STATUS],10000000B ; SEE IF A WRITE OPERATION
                            <1>         ;JZ     short ISNT_WRITE    ; IF NOT, DO NOT ENFORCE ANY VALUES
                            <1>         ;OR     AH,AH               ; CHECK FOR ANY WAIT?
                            <1>         ;JNZ    short DO_WAT        ; IF THERE DO NOT ENFORCE
00004FA2 741E               <1>         jz      short HW_DONE
00004FA4 B40F               <1>         MOV     AH,HD12_SETTLE              ; LOAD 1.2M HEAD SETTLE MINIMUM
00004FA6 8A87[25820100]     <1>         MOV     AL,[DSK_STATE+eDI]  ; LOAD STATE
00004FAC 24C0               <1>         AND     AL,RATE_MSK         ; KEEP ONLY RATE
00004FAE 3C80               <1>         CMP     AL,RATE_250         ; 1.2 M DRIVE ?
00004FB0 7502               <1>         JNZ     short DO_WAT        ; DEFAULT HEAD SETTLE LOADED
                            <1> ;GP3:
00004FB2 B414               <1>         MOV     AH,HD320_SETTLE             ; USE 320/360 HEAD SETTLE
                            <1> ;       JMP     SHORT DO_WAT
                            <1>
                            <1> ;ISNT_WRITE:
                            <1> ;       OR      AH,AH               ; CHECK FOR NO WAIT
                            <1> ;       JZ      short HW_DONE       ; IF NOT WRITE AND 0 ITS OK
                            <1>
                            <1> ;-----      AH CONTAINS NUMBER OF MILLISECONDS TO WAIT
                            <1> DO_WAT:
                            <1> ;       MOV     AL,AH               ; AL = # MILLISECONDS
                            <1> ;       ;XOR    AH,AH               ; AX = # MILLISECONDS
                            <1> J29:                                ;       1 MILLISECOND LOOP
                            <1>         ;mov    cx, WAIT_FDU_HEAD_SETTLE ; 33 ; 1 ms in 30 micro units.
00004FB4 B942000000         <1>         MOV     eCX,66              ; COUNT AT 15.085737 US PER COUNT
00004FB9 E87ED4FFFF         <1>         CALL    WAITF               ; DELAY FOR 1 MILLISECOND
                            <1>         ;DEC    AL                  ; DECREMENT THE COUNT
00004FBE FECC               <1>         dec     ah
00004FC0 75F2               <1>         JNZ     short J29           ; DO AL MILLISECOND # OF TIMES
                            <1> HW_DONE:
00004FC2 C3                 <1>         RETn
                            <1>
                            <1> ;--------------------------------------------------------------------------
                            <1> ; NEC_OUTPUT
                            <1> ;       THIS ROUTINE SENDS A BYTE TO THE NEC CONTROLLER AFTER TESTING
                            <1> ;       FOR CORRECT DIRECTION AND CONTROLLER READY THIS ROUTINE WILL
                            <1> ;       TIME OUT IF THE BYTE IS NOT ACCEPTED WITHIN A REASONABLE AMOUNT
                            <1> ;       OF TIME, SETTING THE DISKETTE STATUS ON COMPLETION.
                            <1> ;
                            <1> ; ON ENTRY:  AH = BYTE TO BE OUTPUT
                            <1> ;
                            <1> ; ON EXIT:   CY = 0  SUCCESS
                            <1> ;            CY = 1  FAILURE -- DISKETTE STATUS UPDATED
                            <1> ;                    IF A FAILURE HAS OCCURRED, THE RETURN IS MADE ONE LEVEL
                            <1> ;                    HIGHER THAN THE CALLER OF NEC OUTPUT. THIS REMOVES THE
                            <1> ;                    REQUIREMENT OF TESTING AFTER EVERY CALL OF NEC_OUTPUT.
                            <1> ;            AX,CX,DX DESTROYED
                            <1> ;--------------------------------------------------------------------------
                            <1>
                            <1> ; 09/12/2014 [Erdogan Tan]
                            <1> ;       (from 'PS2 Hardware Interface Tech. Ref. May 88', Page 09-05.)
                            <1> ; Diskette Drive Controller Status Register (3F4h)
                            <1> ;       This read only register facilitates the transfer of data between
                            <1> ;       the system microprocessor and the controller.
                            <1> ; Bit 7 - When set to 1, the Data register is ready to transfer data
                            <1> ;         with the system micrprocessor.
                            <1> ; Bit 6 - The direction of data transfer. If this bit is set to 0,
                            <1> ;         the transfer is to the controller.
                            <1> ; Bit 5 - When this bit is set to 1, the controller is in the non-DMA mode.
                            <1> ; Bit 4 - When this bit is set to 1, a Read or Write command is being executed.
                            <1> ; Bit 3 - Reserved.
                            <1> ; Bit 2 - Reserved.
                            <1> ; Bit 1 - When this bit is set to 1, dskette drive 1 is in the seek mode.
                            <1> ; Bit 0 - When this bit is set to 1, dskette drive 1 is in the seek mode.
                            <1>
                            <1> ; Data Register (3F5h)
                            <1> ; This read/write register passes data, commands and parameters, and provides
                            <1> ; diskette status information.
                            <1>
                            <1> NEC_OUTPUT:
                            <1>         ;PUSH  BX                   ; SAVE REG.
00004FC3 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.
```

```
2599                            <1>       ;      AL=pattern to look for.
2600                            <1>       ;      DX=Port to test for
2601                            <1>       ;      BH:CX=Number of memory refresh periods to delay.
2602                            <1>       ;           (normally 30 microseconds per period.)
2603                            <1>       ;
2604                            <1>       ;WFP_SHORT:
2605                            <1>       ;      Wait for port if refresh cycle is short (15-80 Us range).
2606                            <1>       ;
2607                            <1>
2608                            <1> ;     mov    bl, WAIT_FDU_SEND_HI+1    ; 0+1
2609                            <1> ;     mov    cx, WAIT_FDU_SEND_LO      ; 16667
2610 00004FC7 B91B410000      <1>       mov    ecx, WAIT_FDU_SEND_LH   ; 16667 (27/02/2015)
2611                            <1> ;
2612                            <1> ;WFPS_OUTER_LP:
2613                            <1> ;     ;
2614                            <1> ;WFPS_CHECK_PORT:
2615                            <1> J23:
2616 00004FCC EC               <1>       IN     AL,DX              ; GET STATUS
2617 00004FCD 24C0             <1>       AND    AL,11000000B       ; KEEP STATUS AND DIRECTION
2618 00004FCF 3C80             <1>       CMP    AL,10000000B       ; STATUS 1 AND DIRECTION 0 ?
2619 00004FD1 7418             <1>       JZ     short J27          ; STATUS AND DIRECTION OK
2620                            <1> WFPS_HI:
2621 00004FD3 E461             <1>       IN     AL, PORT_B   ;061h  ; SYS1 ; wait for hi to lo
2622 00004FD5 A810             <1>       TEST   AL,010H                   ; transition on memory
2623 00004FD7 75FA             <1>       JNZ    SHORT WFPS_HI       ; refresh.
2624                            <1> WFPS_LO:
2625 00004FD9 E461             <1>       IN     AL, PORT_B          ; SYS1
2626 00004FDB A810             <1>       TEST   AL,010H
2627 00004FDD 74FA             <1>       JZ     SHORT WFPS_LO
2628                            <1>       ;LOOP  SHORT WFPS_CHECK_PORT
2629 00004FDF E2EB             <1>       loop   J23    ; 27/02/2015
2630                            <1> ;     ;
2631                            <1> ;     dec    bl
2632                            <1> ;     jnz    short WFPS_OUTER_LP
2633                            <1> ;     jmp    short WFPS_TIMEOUT  ; fail
2634                            <1> ;J23:
2635                            <1> ;     IN     AL,DX              ; GET STATUS
2636                            <1> ;     AND    AL,11000000B       ; KEEP STATUS AND DIRECTION
2637                            <1> ;     CMP    AL,10000000B       ; STATUS 1 AND DIRECTION 0 ?
2638                            <1> ;     JZ     short J27          ; STATUS AND DIRECTION OK
2639                            <1>       ;LOOP  J23                ; CONTINUE TILL CX EXHAUSTED
2640                            <1>       ;DEC   BL                 ; DECREMENT COUNTER
2641                            <1>       ;JNZ   short J23          ; REPEAT TILL DELAY FINISHED, CX = 0
2642                            <1>
2643                            <1>       ;;27/02/2015
2644                            <1>       ;16/12/2014
2645                            <1>         ;;cmp     byte [wait_count], 10   ; (10/18.2 seconds)
2646                            <1>       ;;jb   short J23
2647                            <1>
2648                            <1> ;WFPS_TIMEOUT:
2649                            <1>
2650                            <1> ;-----      FALL THRU TO ERROR RETURN
2651                            <1>
2652 00004FE1 800D[18820100]80 <1>       OR     byte [DSKETTE_STATUS],TIME_OUT
2653                            <1>       ;POP   BX                 ; RESTORE REG.
2654 00004FE8 58               <1>       POP    eAX ; 08/02/2015   ; DISCARD THE RETURN ADDRESS
2655 00004FE9 F9               <1>       STC                       ; INDICATE ERROR TO CALLER
2656 00004FEA C3               <1>       RETn
2657                            <1>
2658                            <1> ;-----      DIRECTION AND STATUS OK; OUTPUT BYTE
2659                            <1>
2660                            <1> J27:
2661 00004FEB 88E0             <1>       MOV    AL,AH              ; GET BYTE TO OUTPUT
2662 00004FED 6642             <1>       INC    DX                 ; DATA PORT = STATUS PORT + 1
2663 00004FEF EE               <1>       OUT    DX,AL              ; OUTPUT THE BYTE
2664                            <1>       ;;NEWIODELAY  ;; 27/02/2015
2665                            <1>       ; 27/02/2015
2666 00004FF0 9C               <1>       PUSHF                     ; SAVE FLAGS
2667 00004FF1 B903000000       <1>       MOV    eCX, 3             ; 30 TO 45 MICROSECONDS WAIT FOR
2668 00004FF6 E841D4FFFF       <1>       CALL   WAITF              ; NEC FLAGS UPDATE CYCLE
2669 00004FFB 9D               <1>       POPF                      ; RESTORE FLAGS FOR EXIT
2670                            <1>       ;POP   BX                 ; RESTORE REG
2671 00004FFC C3               <1>       RETn                      ; CY = 0 FROM TEST INSTRUCTION
2672                            <1>
2673                            <1> ;--------------------------------------------------------------------------
2674                            <1> ; SEEK
2675                            <1> ;     THIS ROUTINE WILL MOVE THE HEAD ON THE NAMED DRIVE TO THE NAMED
2676                            <1> ;     TRACK. IF THE DRIVE HAS NOT BEEN ACCESSED SINCE THE DRIVE
2677                            <1> ;     RESET COMMAND WAS ISSUED, THE DRIVE WILL BE RECALIBRATED.
2678                            <1> ;
2679                            <1> ; ON ENTRY:  DI = DRIVE #
2680                            <1> ;            CH = TRACK #
2681                            <1> ;
2682                            <1> ; ON EXIT:   @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION.
2683                            <1> ;            AX,BX,CX DX DESTROYED
2684                            <1> ;--------------------------------------------------------------------------
2685                            <1> SEEK:
2686 00004FFD 89FB             <1>       MOV    eBX,eDI                   ; BX = DRIVE #
2687 00004FFF B001             <1>       MOV    AL,1               ; ESTABLISH MASK FOR RECALIBRATE TEST
2688 00005001 86CB             <1>       XCHG   CL,BL              ; SET DRIVE VALULE INTO CL
2689 00005003 D2C0             <1>       ROL    AL,CL              ; SHIFT MASK BY THE DRIVE VALUE
2690 00005005 86CB             <1>       XCHG   CL,BL              ; RECOVER TRACK VALUE
2691 00005007 8405[15820100]   <1>       TEST   AL,[SEEK_STATUS]   ; TEST FOR RECALIBRATE REQUIRED
2692 0000500D 7526             <1>       JNZ    short J28A         ; JUMP IF RECALIBRATE NOT REQUIRED
2693                            <1>
2694 0000500F 0805[15820100]   <1>       OR     [SEEK_STATUS],AL   ; TURN ON THE NO RECALIBRATE BIT IN FLAG
2695 00005015 E862000000       <1>       CALL   RECAL              ; RECALIBRATE DRIVE
2696 0000501A 730E             <1>       JNC    short AFT_RECAL          ; RECALIBRATE DONE
2697                            <1>
2698                            <1> ;-----      ISSUE RECALIBRATE FOR 80 TRACK DISKETTES
2699                            <1>
2700 0000501C C605[18820100]00 <1>       MOV    byte [DSKETTE_STATUS],0   ; CLEAR OUT INVALID STATUS
2701 00005023 E854000000       <1>       CALL   RECAL              ; RECALIBRATE DRIVE
2702 00005028 7251             <1>       JC     short RB           ; IF RECALIBRATE FAILS TWICE THEN ERROR
2703                            <1>
```

```
                            <1> AFT_RECAL:
0000502A C687[29820100]00   <1>         MOV     byte [DSK_TRK+eDI],0    ; SAVE NEW CYLINDER AS PRESENT POSITION
00005031 08ED               <1>       OR    CH,CH               ; CHECK FOR SEEK TO TRACK 0
00005033 743F               <1>       JZ    short DO_WAIT       ; HEAD SETTLE, CY = 0 IF JUMP
                            <1>
                            <1> ;-----      DRIVE IS IN SYNCHRONIZATION WITH CONTROLLER, SEEK TO TRACK
                            <1>
00005035 F687[25820100]20   <1> J28A: TEST  byte [DSK_STATE+eDI],DBL_STEP ; CHECK FOR DOUBLE STEP REQUIRED
0000503C 7402               <1>       JZ    short _R7           ; SINGLE STEP REQUIRED BYPASS DOUBLE
0000503E D0E5               <1>       SHL   CH,1                ; DOUBLE NUMBER OF STEP TO TAKE
                            <1>
00005040 3AAF[29820100]     <1> _R7:  CMP   CH, [DSK_TRK+eDI]   ; SEE IF ALREADY AT THE DESIRED TRACK
00005046 7433               <1>       JE    short RB            ; IF YES, DO NOT NEED TO SEEK
                            <1>
00005048 BA[7B500000]       <1>       MOV   eDX, NEC_ERR        ; LOAD RETURN ADDRESS
0000504D 52                 <1>       PUSH  eDX ; (*)           ; ON STACK FOR NEC OUTPUT ERROR
0000504E 88AF[29820100]     <1>       MOV   [DSK_TRK+eDI],CH    ; SAVE NEW CYLINDER AS PRESENT POSITION
00005054 B40F               <1>       MOV   AH,0FH              ; SEEK COMMAND TO NEC
00005056 E868FFFFFF         <1>       CALL  NEC_OUTPUT
0000505B 89FB               <1>       MOV   eBX,eDI                   ; BX = DRIVE #
0000505D 88DC               <1>       MOV   AH,BL               ; OUTPUT DRIVE NUMBER
0000505F E85FFFFFFF         <1>       CALL  NEC_OUTPUT
00005064 8AA7[29820100]     <1>       MOV   AH, [DSK_TRK+eDI]   ; GET CYLINDER NUMBER
0000506A E854FFFFFF         <1>       CALL  NEC_OUTPUT
0000506F E829000000         <1>       CALL  CHK_STAT_2          ; ENDING INTERRUPT AND SENSE STATUS
                            <1>
                            <1> ;-----      WAIT FOR HEAD SETTLE
                            <1>
                            <1> DO_WAIT:
00005074 9C                 <1>       PUSHF                     ; SAVE STATUS
00005075 E816FFFFFF         <1>       CALL  HD_WAIT                   ; WAIT FOR HEAD SETTLE TIME
0000507A 9D                 <1>       POPF                      ; RESTORE STATUS
                            <1> RB:
                            <1> NEC_ERR:
                            <1>       ; 08/02/2015 (code trick here from original IBM PC/AT DISKETTE.ASM)
                            <1>       ; (*) nec_err -> retn (push edx -> pop edx) -> nec_err -> retn
0000507B C3                 <1>       RETn                      ; RETURN TO CALLER
                            <1>
                            <1> ;--------------------------------------------------------------------------
                            <1> ; RECAL
                            <1> ;     RECALIBRATE DRIVE
                            <1> ;
                            <1> ; ON ENTRY:  DI = DRIVE #
                            <1> ;
                            <1> ; ON EXIT:  CY REFLECTS STATUS OF OPERATION.
                            <1> ;--------------------------------------------------------------------------
                            <1> RECAL:
0000507C 6651               <1>       PUSH  CX
0000507E B8[9A500000]       <1>       MOV   eAX, RC_BACK        ; LOAD NEC_OUTPUT ERROR
00005083 50                 <1>       PUSH  eAX
00005084 B407               <1>       MOV   AH,07H              ; RECALIBRATE COMMAND
00005086 E838FFFFFF         <1>       CALL  NEC_OUTPUT
0000508B 89FB               <1>       MOV   eBX,eDI                   ; BX = DRIVE #
0000508D 88DC               <1>       MOV   AH,BL
0000508F E82FFFFFFF         <1>       CALL  NEC_OUTPUT          ; OUTPUT THE DRIVE NUMBER
00005094 E804000000         <1>       CALL  CHK_STAT_2          ; GET THE INTERRUPT AND SENSE INT STATUS
00005099 58                 <1>       POP   eAX                 ; THROW AWAY ERROR
                            <1> RC_BACK:
0000509A 6659               <1>       POP   CX
0000509C C3                 <1>       RETn
                            <1>
                            <1> ;--------------------------------------------------------------------------
                            <1> ; CHK_STAT_2
                            <1> ;     THIS ROUTINE HANDLES THE INTERRUPT RECEIVED AFTER RECALIBRATE,
                            <1> ;     OR SEEK TO THE ADAPTER. THE INTERRUPT IS WAITED FOR, THE
                            <1> ;     INTERRUPT STATUS SENSED, AND THE RESULT RETURNED TO THE CALLER.
                            <1> ;
                            <1> ; ON EXIT:  @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION.
                            <1> ;--------------------------------------------------------------------------
                            <1> CHK_STAT_2:
0000509D B8[C5500000]       <1>         MOV     eAX, CS_BACK            ; LOAD NEC_OUTPUT ERROR ADDRESS
000050A2 50                 <1>       PUSH  eAX
000050A3 E828000000         <1>       CALL  WAIT_INT            ; WAIT FOR THE INTERRUPT
000050A8 721A               <1>       JC    short J34           ; IF ERROR, RETURN IT
000050AA B408               <1>       MOV   AH,08H              ; SENSE INTERRUPT STATUS COMMAND
000050AC E812FFFFFF         <1>       CALL  NEC_OUTPUT
000050B1 E84A000000         <1>       CALL  RESULTS                   ; READ IN THE RESULTS
000050B6 720C               <1>       JC    short J34
000050B8 A0[19820100]       <1>       MOV   AL,[NEC_STATUS]           ; GET THE FIRST STATUS BYTE
000050BD 2460               <1>       AND   AL,01100000B        ; ISOLATE THE BITS
000050BF 3C60               <1>       CMP   AL,01100000B        ; TEST FOR CORRECT VALUE
000050C1 7403               <1>       JZ    short J35           ; IF ERROR, GO MARK IT
000050C3 F8                 <1>       CLC                       ; GOOD RETURN
                            <1> J34:
000050C4 58                 <1>       POP   eAX                 ; THROW AWAY ERROR RETURN
                            <1> CS_BACK:
000050C5 C3                 <1>       RETn
                            <1> J35:
000050C6 800D[18820100]40   <1>       OR    byte [DSKETTE_STATUS], BAD_SEEK
000050CD F9                 <1>       STC                       ; ERROR RETURN CODE
000050CE EBF4               <1>       JMP   SHORT J34
                            <1>
                            <1> ;--------------------------------------------------------------------------
                            <1> ; WAIT_INT
                            <1> ;     THIS ROUTINE WAITS FOR AN INTERRUPT TO OCCUR A TIME OUT ROUTINE
                            <1> ;     TAKES PLACE DURING THE WAIT, SO THAT AN ERROR MAY BE RETURNED
                            <1> ;     IF THE DRIVE IS NOT READY.
                            <1> ;
                            <1> ; ON EXIT:  @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION.
                            <1> ;--------------------------------------------------------------------------
                            <1>
                            <1> ; 17/12/2014
                            <1> ; 2.5 seconds waiting !
                            <1> ;(AWARD BIOS - 1999, WAIT_FDU_INT_LOW, WAIT_FDU_INT_HI)
                            <1> ; amount of time to wait for completion interrupt from NEC.
```

```
2809                                    <1>
2810                                    <1>
2811                                    <1> WAIT_INT:
2812 000050D0 FB                       <1>         STI                         ; TURN ON INTERRUPTS, JUST IN CASE
2813 000050D1 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 000050D2 B986450100               <1>         mov    ecx, WAIT_FDU_INT_LH      ; 83334 (2.5 seconds)
2834                                    <1> WFMS_CHECK_MEM:
2835 000050D7 F605[15820100]80         <1>         test   byte [SEEK_STATUS],INT_FLAG ; TEST FOR INTERRUPT OCCURRING
2836 000050DE 7516                     <1>           jnz     short J37
2837                                    <1> WFMS_HI:
2838 000050E0 E461                     <1>         IN     AL,PORT_B  ; 061h   ; SYS1, wait for lo to hi
2839 000050E2 A810                     <1>         TEST   AL,010H                   ; transition on memory
2840 000050E4 75FA                     <1>         JNZ    SHORT WFMS_HI       ; refresh.
2841                                    <1> WFMS_LO:
2842 000050E6 E461                     <1>         IN     AL,PORT_B           ;SYS1
2843 000050E8 A810                     <1>         TEST   AL,010H
2844 000050EA 74FA                     <1>         JZ     SHORT WFMS_LO
2845 000050EC 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 000050EE 800D[18820100]80         <1>         OR     byte [DSKETTE_STATUS], TIME_OUT ; NOTHING HAPPENED
2869 000050F5 F9                       <1>         STC                         ; ERROR RETURN
2870                                    <1> J37:
2871 000050F6 9C                       <1>         PUSHF                       ; SAVE CURRENT CARRY
2872 000050F7 8025[15820100]7F         <1>         AND    byte [SEEK_STATUS], ~INT_FLAG ; TURN OFF INTERRUPT FLAG
2873 000050FE 9D                       <1>         POPF                        ; RECOVER CARRY
2874 000050FF 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 00005100 57                       <1>         PUSH   eDI
2886 00005101 BF[19820100]             <1>         MOV    eDI, NEC_STATUS           ; POINTER TO DATA AREA
2887 00005106 B307                     <1>         MOV    BL,7                ; MAX STATUS BYTES
2888 00005108 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
2897                                    <1>
2898                                    <1>         ;Time to wait while waiting for each byte of NEC results = .5
2899                                    <1>         ;seconds.  .5 seconds = 500,000 micros.  500,000/30 = 16,667.
2900                                    <1>         ; 27/02/2015
2901 0000510C B91B410000               <1>         mov    ecx, WAIT_FDU_RESULTS_LH ; 16667
2902                                    <1>         ;mov   cx, WAIT_FDU_RESULTS_LO  ; 16667
2903                                    <1>         ;mov   bh, WAIT_FDU_RESULTS_HI+1 ; 0+1
2904                                    <1>
2905                                    <1> WFPSR_OUTER_LP:
2906                                    <1>         ;
2907                                    <1> WFPSR_CHECK_PORT:
2908                                    <1> J39:                                ; WAIT FOR MASTER
2909 00005111 EC                       <1>         IN     AL,DX               ; GET STATUS
2910 00005112 24C0                     <1>         AND    AL,11000000B        ; KEEP ONLY STATUS AND DIRECTION
2911 00005114 3CC0                     <1>         CMP    AL,11000000B        ; STATUS 1 AND DIRECTION 1 ?
2912 00005116 7418                     <1>         JZ     short J42           ; STATUS AND DIRECTION OK
2913                                    <1> WFPSR_HI:
```

```
                                <1>       IN    AL, PORT_B   ;061h  ; SYS1 ; wait for hi to lo
                                <1>       TEST  AL,010H                ; transition on memory
                                <1>       JNZ   SHORT WFPSR_HI         ; refresh.
                                <1> WFPSR_LO:
                                <1>       IN    AL, PORT_B         ; SYS1
                                <1>       TEST  AL,010H
                                <1>       JZ    SHORT WFPSR_LO
                                <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:
00005126 800D[18820100]80       <1>       OR    byte [DSKETTE_STATUS],TIME_OUT
0000512D F9                     <1>       STC                          ; SET ERROR RETURN
0000512E EB29                   <1>       JMP   SHORT POPRES           ; POP REGISTERS AND RETURN
                                <1>
                                <1> ;-----      READ IN THE STATUS
                                <1>
                                <1> J42:
00005130 EB00                   <1>       JMP   $+2                    ; I/O DELAY
00005132 6642                   <1>       INC   DX                     ; POINT AT DATA PORT
00005134 EC                     <1>       IN    AL,DX                  ; GET THE DATA
                                <1>       ; 16/12/2014
                                <1>       NEWIODELAY
00005135 E6EB                   <2>  out 0ebh,al
00005137 8807                   <1>         MOV     [eDI],AL                ; STORE THE BYTE
00005139 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>
0000513A B903000000             <1>       MOV   eCX,3                  ; MINIMUM 24 MICROSECONDS FOR NEC
0000513F E8F8D2FFFF             <1>       CALL  WAITF                  ; WAIT 30 TO 45 MICROSECONDS
00005144 664A                   <1>       DEC   DX                     ; POINT AT STATUS PORT
00005146 EC                     <1>       IN    AL,DX                  ; GET STATUS
                                <1>       ; 16/12/2014
                                <1>       NEWIODELAY
00005147 E6EB                   <2>  out 0ebh,al
                                <1>       ;
00005149 A810                   <1>       TEST  AL,00010000B           ; TEST FOR NEC STILL BUSY
0000514B 740C                   <1>       JZ    short POPRES           ; RESULTS DONE ?
                                <1>
0000514D FECB                   <1>       DEC   BL                     ; DECREMENT THE STATUS COUNTER
0000514F 75BB                   <1>         JNZ     short _R10              ; GO BACK FOR MORE
00005151 800D[18820100]20       <1>       OR    byte [DSKETTE_STATUS],BAD_NEC ; TOO MANY STATUS BYTES
00005158 F9                     <1>       STC                          ; SET ERROR FLAG
                                <1>
                                <1> ;-----      RESULT OPERATION IS DONE
                                <1> POPRES:
00005159 5F                     <1>       POP   eDI
0000515A 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:
0000515B E8A2FDFFFF             <1>       CALL  MOTOR_ON               ; TURN ON THE MOTOR IF OFF
00005160 66BAF703               <1>       MOV   DX,03F7H               ; ADDRESS DIGITAL INPUT REGISTER
00005164 EC                     <1>       IN    AL,DX                  ; INPUT DIGITAL INPUT REGISTER
00005165 A880                   <1>       TEST  AL,DSK_CHG             ; CHECK FOR DISK CHANGE LINE ACTIVE
00005167 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:
00005168 E895FDFFFF             <1>       CALL  MOTOR_ON               ; TURN ON MOTOR IF NOT ALREADY ON
0000516D E80AFFFFFF             <1>       CALL  RECAL                  ; RECALIBRATE DRIVE
00005172 7251                   <1>       JC    short DD_BAC           ; ASSUME NO DRIVE PRESENT
00005174 B530                   <1>       MOV   CH,TRK_SLAP            ; SEEK TO TRACK 48
00005176 E882FEFFFF             <1>       CALL  SEEK
0000517B 7248                   <1>       JC    short DD_BAC           ; ERROR NO DRIVE
0000517D B50B                   <1>       MOV   CH,QUIET_SEEK+1              ; SEEK TO TRACK 10
                                <1> SK_GIN:
0000517F FECD                   <1>       DEC   CH                     ; DECREMENT TO NEXT TRACK
```

```
3017 00005181 6651                <1>       PUSH   CX                 ; SAVE TRACK
3018 00005183 E875FEFFFF          <1>       CALL   SEEK
3019 00005188 723C                <1>       JC     short POP_BAC       ; POP AND RETURN
3020 0000518A B8[C6510000]        <1>       MOV    eAX, POP_BAC        ; LOAD NEC OUTPUT ERROR ADDRESS
3021 0000518F 50                  <1>       PUSH   eAX
3022 00005190 B404                <1>       MOV    AH,SENSE_DRV_ST           ; SENSE DRIVE STATUS COMMAND BYTE
3023 00005192 E82CFEFFFF          <1>       CALL   NEC_OUTPUT          ; OUTPUT TO NEC
3024 00005197 6689F8              <1>       MOV    AX,DI               ; AL = DRIVE
3025 0000519A 88C4                <1>       MOV    AH,AL               ; AH = DRIVE
3026 0000519C E822FEFFFF          <1>       CALL   NEC_OUTPUT          ; OUTPUT TO NEC
3027 000051A1 E85AFFFFFF          <1>       CALL   RESULTS                   ; GO GET STATUS
3028 000051A6 58                  <1>       POP    eAX                 ; THROW AWAY ERROR ADDRESS
3029 000051A7 6659                <1>       POP    CX                  ; RESTORE TRACK
3030 000051A9 F605[19820100]10    <1>       TEST   byte [NEC_STATUS], HOME    ; TRACK 0 ?
3031 000051B0 74CD                <1>       JZ     short SK_GIN        ; GO TILL TRACK 0
3032 000051B2 08ED                <1>       OR     CH,CH               ; IS HOME AT TRACK 0
3033 000051B4 7408                <1>       JZ     short IS_80         ; MUST BE 80 TRACK DRIVE
3034                              <1>
3035                              <1> ;     DRIVE IS A 360; SET DRIVE TO DETERMINED;
3036                              <1> ;     SET MEDIA TO DETERMINED AT RATE 250.
3037                              <1>
3038 000051B6 808F[25820100]94    <1>       OR     byte [DSK_STATE+eDI], DRV_DET+MED_DET+RATE_250
3039 000051BD C3                  <1>       RETn                       ; ALL INFORMATION SET
3040                              <1> IS_80:
3041 000051BE 808F[25820100]01    <1>       OR     byte [DSK_STATE+eDI], TRK_CAPA ; SETUP 80 TRACK CAPABILITY
3042                              <1> DD_BAC:
3043 000051C5 C3                  <1>       RETn
3044                              <1> POP_BAC:
3045 000051C6 6659                <1>       POP    CX                  ; THROW AWAY
3046 000051C8 C3                  <1>       RETn
3047                              <1>
3048                              <1> fdc_int:
3049                              <1>         ; 30/07/2015
3050                              <1>         ; 16/02/2015
3051                              <1> ;int_0Eh: ; 11/12/2014
3052                              <1>
3053                              <1> ;--- HARDWARE INT 0EH -- ( IRQ LEVEL  6 ) -------------------------------------
3054                              <1> ; DISK_INT
3055                              <1> ;     THIS ROUTINE HANDLES THE DISKETTE INTERRUPT.
3056                              <1> ;
3057                              <1> ; ON EXIT:  THE INTERRUPT FLAG IS SET IN @SEEK_STATUS.
3058                              <1> ;--------------------------------------------------------------------------------
3059                              <1> DISK_INT_1:
3060                              <1>
3061 000051C9 6650                <1>       PUSH   AX                  ; SAVE WORK REGISTER
3062 000051CB 1E                  <1>       push   ds
3063 000051CC 66B81000            <1>       mov    ax, KDATA
3064 000051D0 8ED8                <1>       mov    ds, ax
3065 000051D2 800D[15820100]80    <1>         OR      byte [SEEK_STATUS], INT_FLAG ; TURN ON INTERRUPT OCCURRED
3066 000051D9 B020                <1>       MOV     AL,EOI                   ; END OF INTERRUPT MARKER
3067 000051DB E620                <1>       OUT    INTA00,AL           ; INTERRUPT CONTROL PORT
3068 000051DD 1F                  <1>       pop    ds
3069 000051DE 6658                <1>       POP    AX                  ; RECOVER REGISTER
3070 000051E0 CF                  <1>       IRETd                      ; RETURN FROM INTERRUPT
3071                              <1>
3072                              <1> ;--------------------------------------------------------------------------------
3073                              <1> ; DSKETTE_SETUP
3074                              <1> ;     THIS ROUTINE DOES A PRELIMINARY CHECK TO SEE WHAT TYPE OF
3075                              <1> ;     DISKETTE DRIVES ARE ATTACH TO THE SYSTEM.
3076                              <1> ;--------------------------------------------------------------------------------
3077                              <1>
3078                              <1> ; 29/05/2016 - TRDOS 386 (TRDOS v2.0)
3079                              <1>
3080                              <1> DSKETTE_SETUP:
3081                              <1>       ;PUSH  AX                  ; SAVE REGISTERS
3082                              <1>       ;PUSH  BX
3083                              <1>       ;PUSH  CX
3084 000051E1 52                  <1>       PUSH   eDX
3085                              <1>       ;PUSH  DI
3086                              <1>       ;;PUSH DS
3087                              <1>       ; 14/12/2014
3088                              <1>       ;mov   word [DISK_POINTER], MD_TBL6
3089                              <1>       ;mov   [DISK_POINTER+2], cs
3090                              <1>       ;
3091                              <1>       ;OR    byte [RTC_WAIT_FLAG], 1    ; NO RTC WAIT, FORCE USE OF LOOP
3092 000051E2 31FF                <1>       XOR    eDI,eDI                    ; INITIALIZE DRIVE POINTER
3093 000051E4 66C705[25820100]00- <1>       MOV    WORD [DSK_STATE],0  ; INITIALIZE STATES
3093 000051EC 00                  <1>
3094 000051ED 8025[20820100]33    <1>       AND    byte [LASTRATE],~(STRT_MSK+SEND_MSK) ; CLEAR START & SEND
3095 000051F4 800D[20820100]C0    <1>       OR     byte [LASTRATE],SEND_MSK ; INITIALIZE SENT TO IMPOSSIBLE
3096 000051FB C605[15820100]00    <1>       MOV    byte [SEEK_STATUS],0       ; INDICATE RECALIBRATE NEEDED
3097 00005202 C605[17820100]00    <1>       MOV    byte [MOTOR_COUNT],0       ; INITIALIZE MOTOR COUNT
3098 00005209 C605[16820100]00    <1>       MOV    byte [MOTOR_STATUS],0      ; INITIALIZE DRIVES TO OFF STATE
3099 00005210 C605[18820100]00    <1>       MOV    byte [DSKETTE_STATUS],0    ; NO ERRORS
3100                              <1>       ;
3101                              <1>       ; 28/02/2015
3102                              <1>       ;mov   word [cfd], 100h
3103 00005217 E82BF2FFFF          <1>       call   DSK_RESET
3104 0000521C 5A                  <1>       pop    edx
3105 0000521D F8                  <1>       clc    ; 29/05/2016
3106 0000521E 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
```

```
                                <1> ;sup1:
                                <1> ;       MOV    byte [SEEK_STATUS],0      ; FORCE RECALIBRATE
                                <1> ;       ;AND   byte [RTC_WAIT_FLAG],0FEH ; ALLOW FOR RTC WAIT
                                <1> ;       CALL   SETUP_END           ; VARIOUS CLEANUPS
                                <1> ;       ;;POP  DS                  ; RESTORE CALLERS REGISTERS
                                <1> ;       ;POP   DI
                                <1> ;       POP    eDX
                                <1> ;       ;POP   CX
                                <1> ;       ;POP   BX
                                <1> ;       ;POP   AX
                                <1> ;       RETn
                                <1>
                                <1> ;////////////////////////////////////////////////////
                                <1> ;; END OF DISKETTE I/O ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                                <1> ;
                                <1>
                                <1> int13h: ; 21/02/2015
0000521F F8                     <1>         clc ; 20/07/2020
00005220 9C                     <1>         pushfd
00005221 0E                     <1>         push   cs
00005222 E848010000             <1>         call   DISK_IO
00005227 C3                     <1>         retn
                                <1>
                                <1> ;;;;;; DISK I/O ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;; 21/02/2015 ;;;
                                <1> ;////////////////////////////////////////////////////////////////////
                                <1>
                                <1> ; DISK I/O - Erdogan Tan (Retro UNIX 386 v1 project)
                                <1> ; 18/02/2016
                                <1> ; 17/02/2016
                                <1> ; 23/02/2015
                                <1> ; 21/02/2015 (unix386.s)
                                <1> ; 22/12/2014 - 14/02/2015 (dsectrm2.s)
                                <1> ;
                                <1> ; Original Source Code:
                                <1> ; DISK ----- 09/25/85 FIXED DISK BIOS
                                <1> ; (IBM PC XT Model 286 System BIOS Source Code, 04-21-86)
                                <1> ;
                                <1> ; Modifications: by reference of AWARD BIOS 1999 (D1A0622)
                                <1> ;             Source Code - ATORGS.ASM, AHDSK.ASM
                                <1> ;
                                <1>
                                <1> ;The wait for controller to be not busy is 10 seconds.
                                <1> ;10,000,000 / 30 = 333,333.  333,333 decimal = 051615h
                                <1> ;;WAIT_HDU_CTLR_BUSY_LO   equ    1615h
                                <1> ;;WAIT_HDU_CTLR_BUSY_HI   equ      05h
                                <1> WAIT_HDU_CTRL_BUSY_LH     equ    51615h  ;21/02/2015
                                <1>
                                <1> ;The wait for controller to issue completion interrupt is 10 seconds.
                                <1> ;10,000,000 / 30 = 333,333.  333,333 decimal = 051615h
                                <1> ;;WAIT_HDU_INT_LO  equ    1615h
                                <1> ;;WAIT_HDU_INT_HI  equ      05h
                                <1> WAIT_HDU_INT_LH           equ    51615h ; 21/02/2015
                                <1>
                                <1> ;The wait for Data request on read and write longs is
                                <1> ;2000 us. (?)
                                <1> ;;WAIT_HDU_DRQ_LO  equ    1000   ; 03E8h
                                <1> ;;WAIT_HDU_DRQ_HI  equ    0
                                <1> WAIT_HDU_DRQ_LH           equ    1000   ; 21/02/2015
                                <1>
                                <1> ; Port 61h (PORT_B)
                                <1> SYS1        equ    61h    ; PORT_B  (diskette.inc)
                                <1>
                                <1> ; 23/12/2014
                                <1> %define CMD_BLOCK       eBP-8  ; 21/02/2015
                                <1>
                                <1> ; 30/08/2020 - TRDOS 386 v2
                                <1>
                                <1> ;--- INT 13H ----------------------------------------------------------------
                                <1> ;                                                                        :
                                <1> ; FIXED DISK I/O INTERFACE                                                   :
                                <1> ;                                                                        :
                                <1> ;     THIS INTERFACE PROVIDES ACCESS TO 5 1/4" FIXED DISKS THROUGH        :
                                <1> ;     THE IBM FIXED DISK CONTROLLER.                                     :
                                <1> ;                                                                        :
                                <1> ;     THE  BIOS  ROUTINES  ARE  MEANT  TO  BE  ACCESSED  THROUGH        :
                                <1> ;     SOFTWARE  INTERRUPTS  ONLY.    ANY  ADDRESSES  PRESENT     IN      :
                                <1> ;     THESE  LISTINGS  ARE  INCLUDED    ONLY  FOR  COMPLETENESS,        :
                                <1> ;     NOT  FOR  REFERENCE.  APPLICATIONS   WHICH  REFERENCE  ANY        :
                                <1> ;     ABSOLUTE  ADDRESSES  WITHIN  THE  CODE SEGMENTS  OF  BIOS        :
                                <1> ;     VIOLATE  THE  STRUCTURE  AND  DESIGN  OF  BIOS.                  :
                                <1> ;                                                                        :
                                <1> ;-------------------------------------------------------------------------:
                                <1> ;                                                                        :
                                <1> ; INPUT  (AH)= HEX COMMAND VALUE                                         :
                                <1> ;                                                                        :
                                <1> ;     (AH)= 00H  RESET DISK (DL = 80H,81H) / DISKETTE                    :
                                <1> ;     (AH)= 01H  READ THE STATUS OF THE LAST DISK OPERATION INTO (AL)     :
                                <1> ;                NOTE: DL < 80H - DISKETTE                               :
                                <1> ;                      DL > 80H - DISK                                   :
                                <1> ;     (AH)= 02H  READ THE DESIRED SECTORS INTO MEMORY                    :
                                <1> ;     (AH)= 03H  WRITE THE DESIRED SECTORS FROM MEMORY                   :
                                <1> ;     (AH)= 04H  VERIFY THE DESIRED SECTORS                              :
                                <1> ;     (AH)= 05H  FORMAT THE DESIRED TRACK                                :
                                <1> ;     (AH)= 06H  UNUSED                                                  :
                                <1> ;     (AH)= 07H  UNUSED                                                  :
                                <1> ;     (AH)= 08H  RETURN THE CURRENT DRIVE PARAMETERS                     :
                                <1> ;     (AH)= 09H  INITIALIZE DRIVE PAIR CHARACTERISTICS                   :
                                <1> ;                INTERRUPT 41 POINTS TO DATA BLOCK FOR DRIVE 0          :
                                <1> ;                INTERRUPT 46 POINTS TO DATA BLOCK FOR DRIVE 1          :
                                <1> ;     (AH)= 0AH  READ LONG                                               :
                                <1> ;     (AH)= 0BH  WRITE LONG  (READ & WRITE LONG ENCOMPASS 512 + 4 BYTES ECC) :
                                <1> ;     (AH)= 0CH  SEEK                                                    :
                                <1> ;     (AH)= 0DH  ALTERNATE DISK RESET (SEE DL)                           :
                                <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>
```

```
<1> ;--------------------------------------------------------
<1> ;                                                      :
<1> ; FIXED DISK PARAMETER TABLE                          :
<1> ;  -  THE TABLE IS COMPOSED OF A BLOCK DEFINED AS:    :
<1> ;                                                      :
<1> ;  +0 (1 WORD) - MAXIMUM NUMBER OF CYLINDERS          :
<1> ;  +2 (1 BYTE) - MAXIMUM NUMBER OF HEADS              :
<1> ;  +3 (1 WORD) - NOT USED/SEE PC-XT                   :
<1> ;  +5 (1 WORD) - STARTING WRITE PRECOMPENSATION CYL   :
<1> ;  +7 (1 BYTE) - MAXIMUM ECC DATA BURST LENGTH        :
<1> ;  +8 (1 BYTE) - CONTROL BYTE                         :
<1> ;               BIT   7 DISABLE RETRIES -OR-         :
<1> ;               BIT   6 DISABLE RETRIES              :
<1> ;               BIT   3 MORE THAN 8 HEADS            :
<1> ;  +9 (3 BYTES)- NOT USED/SEE PC-XT                   :
<1> ; +12 (1 WORD) - LANDING ZONE                         :
<1> ; +14 (1 BYTE) - NUMBER OF SECTORS/TRACK              :
<1> ; +15 (1 BYTE) - RESERVED FOR FUTURE USE              :
<1> ;                                                      :
<1> ;      - TO DYNAMICALLY DEFINE A SET OF PARAMETERS   :
<1> ;        BUILD A TABLE FOR UP TO 15 TYPES AND PLACE  :
<1> ;        THE CORRESPONDING VECTOR INTO INTERRUPT 41  :
<1> ;        FOR DRIVE 0 AND INTERRUPT 46 FOR DRIVE 1.   :
<1> ;                                                      :
<1> ;--------------------------------------------------------
<1>
<1> ;--------------------------------------------------------
<1> ;                                                      :
<1> ; HARDWARE SPECIFIC VALUES                            :
<1> ;                                                      :
<1> ;  -  CONTROLLER I/O PORT                             :
<1> ;                                                      :
<1> ;     > WHEN READ FROM:                               :
<1> ;     HF_PORT+0 - READ DATA (FROM CONTROLLER TO CPU)      :
<1> ;     HF_PORT+1 - GET ERROR REGISTER                  :
<1> ;     HF_PORT+2 - GET SECTOR COUNT                    :
<1> ;     HF_PORT+3 - GET SECTOR NUMBER                   :
<1> ;     HF_PORT+4 - GET CYLINDER LOW                    :
<1> ;     HF_PORT+5 - GET CYLINDER HIGH (2 BITS)          :
<1> ;     HF_PORT+6 - GET SIZE/DRIVE/HEAD              :
<1> ;     HF_PORT+7 - GET STATUS REGISTER              :
<1> ;                                                      :
<1> ;     > WHEN WRITTEN TO:                           :
<1> ;     HF_PORT+0 - WRITE DATA (FROM CPU TO CONTROLLER) :
<1> ;     HF_PORT+1 - SET PRECOMPENSATION CYLINDER         :
<1> ;     HF_PORT+2 - SET SECTOR COUNT                    :
<1> ;     HF_PORT+3 - SET SECTOR NUMBER                   :
<1> ;     HF_PORT+4 - SET CYLINDER LOW                    :
<1> ;     HF_PORT+5 - SET CYLINDER HIGH (2 BITS)          :
<1> ;     HF_PORT+6 - SET SIZE/DRIVE/HEAD              :
<1> ;     HF_PORT+7 - SET COMMAND REGISTER             :
<1> ;                                                      :
<1> ;--------------------------------------------------------
<1>
<1> ;HF_PORT     EQU    01F0H  ; DISK PORT
<1> ;HF1_PORT    equ    0170h
<1> ;HF_REG_PORT EQU    03F6H
<1> ;HF1_REG_PORT       equ    0376h
<1>
<1> HDC1_BASEPORT       equ    1F0h
<1> HDC2_BASEPORT       equ    170h
<1>
<1> align 2
<1>
<1> ;-----              STATUS REGISTER
<1>
<1> ST_ERROR     EQU    00000001B    ;
<1> ST_INDEX     EQU    00000010B    ;
<1> ST_CORRCTD   EQU    00000100B    ; ECC CORRECTION SUCCESSFUL
<1> ST_DRQ              EQU    00001000B    ;
<1> ST_SEEK_COMPL       EQU    00010000B    ; SEEK COMPLETE
<1> ST_WRT_FLT   EQU    00100000B    ; WRITE FAULT
<1> ST_READY     EQU    01000000B    ;
<1> ST_BUSY      EQU    10000000B    ;
<1>
<1> ;-----              ERROR REGISTER
<1>
<1> ERR_DAM      EQU    00000001B    ; DATA ADDRESS MARK NOT FOUND
<1> ERR_TRK_0    EQU    00000010B    ; TRACK 0 NOT FOUND ON RECAL
<1> ERR_ABORT    EQU    00000100B    ; ABORTED COMMAND
<1> ;            EQU    00001000B    ; NOT USED
<1> ERR_ID               EQU    00010000B    ; ID NOT FOUND
<1> ;            EQU    00100000B    ; NOT USED
<1> ERR_DATA_ECC EQU    01000000B
<1> ERR_BAD_BLOCK       EQU    10000000B
<1>
<1>
<1> RECAL_CMD    EQU    00010000B    ; DRIVE RECAL(10H)
<1> READ_CMD     EQU    00100000B    ;      READ   (20H)
<1> WRITE_CMD    EQU    00110000B    ;      WRITE  (30H)
<1> VERIFY_CMD   EQU    01000000B    ;      VERIFY (40H)
<1> FMTTRK_CMD   EQU    01010000B    ; FORMAT TRACK      (50H)
<1> INIT_CMD     EQU    01100000B    ;   INITIALIZE      (60H)
<1> SEEK_CMD     EQU    01110000B    ;      SEEK   (70H)
<1> DIAG_CMD     EQU    10010000B    ; DIAGNOSTIC (90H)
<1> SET_PARM_CMD EQU    10010001B    ; DRIVE PARMS(91H)
<1> NO_RETRIES   EQU    00000001B    ; CHD MODIFIER      (01H)
<1> ECC_MODE     EQU    00000010B    ; CMD MODIFIER      (02H)
<1> BUFFER_MODE  EQU    00001000B    ; CMD MODIFIER      (08H)
<1>
<1> ;MAX_FILE    EQU    2
<1> ;S_MAX_FILE  EQU    2
<1> MAX_FILE     equ    4            ; 22/12/2014
<1> S_MAX_FILE   equ    4            ; 22/12/2014
<1>
```

```
                                <1> DELAY_1      EQU    25H          ; DELAY FOR OPERATION COMPLETE
                                <1> DELAY_2      EQU    0600H        ; DELAY FOR READY
                                <1> DELAY_3      EQU    0100H        ; DELAY FOR DATA REQUEST
                                <1>
                                <1> HF_FAIL      EQU    08H          ; CMOS FLAG IN BYTE 0EH
                                <1>
                                <1> ;-----              COMMAND BLOCK REFERENCE
                                <1>
                                <1> ;CMD_BLOCK      EQU     BP-8            ; @CMD_BLOCK REFERENCES BLOCK HEAD IN SS
                                <1>                                         ;  (BP) POINTS TO COMMAND BLOCK TAIL
                                <1>                                         ;      AS DEFINED BY THE "ENTER" PARMS
                                <1> ; 19/12/2014
                                <1> ORG_VECTOR   equ    4*13h        ; INT 13h vector
                                <1> DISK_VECTOR  equ    4*40h        ; INT 40h vector (for floppy disks)
                                <1> ;HDISK_INT   equ    4*76h        ; Primary HDC - Hardware interrupt (IRQ14)
                                <1> ;HDISK_INT1  equ    4*76h        ; Primary HDC - Hardware interrupt (IRQ14)
                                <1> ;HDISK_INT2  equ    4*77h        ; Secondary HDC - Hardware interrupt (IRQ15)
                                <1> ;HF_TBL_VEC  equ    4*41h        ; Pointer to 1st fixed disk parameter table
                                <1> ;HF1_TBL_VEC equ    4*46h        ; Pointer to 2nd fixed disk parameter table
                                <1>
                                <1> align 2
                                <1>
                                <1> ;--------------------------------------------------------------
                                <1> ; FIXED DISK I/O SETUP                                          :
                                <1> ;                                                               :
                                <1> ;  -  ESTABLISH TRANSFER VECTORS FOR THE FIXED DISK             :
                                <1> ;  -  PERFORM POWER ON DIAGNOSTICS                              :
                                <1> ;     SHOULD AN ERROR OCCUR A "1701" MESSAGE IS DISPLAYED       :
                                <1> ;                                                               :
                                <1> ;--------------------------------------------------------------
                                <1>
                                <1> ; 29/05/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>
                                <1> DISK_SETUP:
                                <1>       ;CLI
                                <1>       ;;MOV  AX,ABS0                  ; GET ABSOLUTE SEGMENT
                                <1>       ;xor   ax,ax
                                <1>       ;MOV   DS,AX                    ; SET SEGMENT REGISTER
                                <1>       ;MOV   AX, [ORG_VECTOR]         ; GET DISKETTE VECTOR
                                <1>       ;MOV   [DISK_VECTOR],AX         ;  INTO INT 40H
                                <1>       ;MOV   AX, [ORG_VECTOR+2]
                                <1>       ;MOV   [DISK_VECTOR+2],AX
                                <1>       ;MOV   word [ORG_VECTOR],DISK_IO ; FIXED DISK HANDLER
                                <1>       ;MOV   [ORG_VECTOR+2],CS
                                <1>       ; 1st controller (primary master, slave)   - IRQ 14
                                <1>       ;;MOV  word [HDISK_INT],HD_INT         ; FIXED DISK INTERRUPT
                                <1>       ;mov   word [HDISK_INT1],HD_INT  ;
                                <1>       ;;MOV  [HDISK_INT+2],CS
                                <1>       ;mov   [HDISK_INT1+2],CS
                                <1>       ; 2nd controller (secondary master, slave) - IRQ 15
                                <1>       ;mov   word [HDISK_INT2],HD1_INT ;
                                <1>       ;mov   [HDISK_INT2+2],CS
                                <1>       ;
                                <1>       ;;MOV  word [HF_TBL_VEC],HD0_DPT ; PARM TABLE DRIVE 80
                                <1>       ;;MOV  word [HF_TBL_VEC+2],DPT_SEGM
                                <1>       ;;MOV  word [HF1_TBL_VEC],HD1_DPT; PARM TABLE DRIVE 81
                                <1>       ;;MOV  word [HF1_TBL_VEC+2],DPT_SEGM
                                <1>       ;push  cs
                                <1>       ;pop   ds
                                <1>       ;mov   word [HDPM_TBL_VEC],HD0_DPT      ; PARM TABLE DRIVE 80h
                                <1>       ;mov   word [HDPM_TBL_VEC+2],DPT_SEGM
00005228 C705[30820100]0000-    <1>       mov    dword [HDPM_TBL_VEC], (DPT_SEGM*16)+HD0_DPT
00005230 0900                   <1>
                                <1>       ;mov   word [HDPS_TBL_VEC],HD1_DPT      ; PARM TABLE DRIVE 81h
                                <1>       ;mov   word [HDPS_TBL_VEC+2],DPT_SEGM
00005232 C705[34820100]2000-    <1>       mov    dword [HDPS_TBL_VEC], (DPT_SEGM*16)+HD1_DPT
0000523A 0900                   <1>
                                <1>       ;mov   word [HDSM_TBL_VEC],HD2_DPT      ; PARM TABLE DRIVE 82h
                                <1>       ;mov   word [HDSM_TBL_VEC+2],DPT_SEGM
0000523C C705[38820100]4000-    <1>       mov    dword [HDSM_TBL_VEC], (DPT_SEGM*16)+HD2_DPT
00005244 0900                   <1>
                                <1>       ;mov   word [HDSS_TBL_VEC],HD3_DPT      ; PARM TABLE DRIVE 83h
                                <1>       ;mov   word [HDSS_TBL_VEC+2],DPT_SEGM
00005246 C705[3C820100]6000-    <1>       mov    dword [HDSS_TBL_VEC], (DPT_SEGM*16)+HD3_DPT
0000524E 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
00005250 BE[F06D0000]           <1>       mov    esi, hd0_type
                                <1>       ;mov   cx, 4
00005255 B904000000             <1>       mov    ecx, 4
                                <1> hde_l:
0000525A AC                     <1>       lodsb
```

```
3536 0000525B 3C80               <1>       cmp    al, 80h                  ; 8?h = existing
3537 0000525D 7206               <1>       jb     short _L4
3538 0000525F FE05[2C820100]     <1>       inc    byte [HF_NUM]       ; + 1 hard (fixed) disk drives
3539                             <1> _L4: ; 26/02/2015
3540 00005265 E2F3               <1>       loop   hde_l
3541                             <1> ;_L4:                          ; 0 <= [HF_NUM] =< 4
3542                             <1> ;L4:
3543                             <1>       ;
3544                             <1>       ;; 31/12/2014 - cancel controller diagnostics here
3545                             <1>       ;;;mov       cx, 3  ; 26/12/2014 (Award BIOS 1999)
3546                             <1>       ;;mov  cl, 3
3547                             <1>       ;;
3548                             <1>       ;;MOV  DL,80H              ; CHECK THE CONTROLLER
3549                             <1> ;;hdc_dl:
3550                             <1>       ;;MOV  AH,14H              ; USE CONTROLLER DIAGNOSTIC COMMAND
3551                             <1>       ;;INT  13H                 ; CALL BIOS WITH DIAGNOSTIC COMMAND
3552                             <1>       ;;;JC  short CTL_ERRX            ; DISPLAY ERROR MESSAGE IF BAD RETURN
3553                             <1>       ;;;jc  short POD_DONE ;22/12/2014
3554                             <1>       ;;jnc  short hdc_reset0
3555                             <1>       ;;loop hdc_dl
3556                             <1>       ;;; 27/12/2014
3557                             <1>       ;;stc
3558                             <1>       ;;retn
3559                             <1>       ;
3560                             <1> ;;hdc_reset0:
3561                             <1>       ; 18/01/2015
3562 00005267 8A0D[2C820100]     <1>       mov    cl, [HF_NUM]
3563 0000526D 20C9               <1>       and    cl, cl
3564 0000526F 740E               <1>       jz     short POD_DONE
3565                             <1>       ;
3566 00005271 B27F               <1>       mov    dl, 7Fh
3567                             <1> hdc_reset1:
3568 00005273 FEC2               <1>       inc    dl
3569                             <1>       ;; 31/12/2015
3570                             <1>       ;;push dx
3571                             <1>       ;;push cx
3572                             <1>       ;;push ds
3573                             <1>       ;;sub  ax, ax
3574                             <1>       ;;mov  ds, ax
3575                             <1>       ;;MOV  AX, [TIMER_LOW]           ; GET START TIMER COUNTS
3576                             <1>       ;;pop  ds
3577                             <1>       ;;MOV  BX,AX
3578                             <1>       ;;ADD  AX,6*182            ; 60 SECONDS* 18.2
3579                             <1>       ;;MOV  CX,AX
3580                             <1>       ;;mov  word [wait_count], 0      ; 22/12/2014 (reset wait counter)
3581                             <1>       ;;
3582                             <1>       ;; 31/12/2014 - cancel HD_RESET_1
3583                             <1>       ;;CALL HD_RESET_1          ; SET UP DRIVE 0, (1,2,3)
3584                             <1>       ;;pop  cx
3585                             <1>       ;;pop  dx
3586                             <1>       ;;
3587                             <1>       ; 18/01/2015
3588 00005275 B40D               <1>       mov    ah, 0Dh ; ALTERNATE RESET
3589                             <1>       ;int   13h
3590 00005277 E8A3FFFFFF         <1>       call   int13h
3591 0000527C E2F5               <1>       loop   hdc_reset1
3592 0000527E F8                 <1>       clc    ; 29/05/2016
3593                             <1> POD_DONE:
3594 0000527F C3                 <1>       RETn
3595                             <1>
3596                             <1> ;;-----      POD_ERROR
3597                             <1>
3598                             <1> ;;CTL_ERRX:
3599                             <1> ;     ;MOV   SI,OFFSET F1782     ; CONTROLLER ERROR
3600                             <1> ;     ;CALL  SET_FAIL            ; DO NOT IPL FROM DISK
3601                             <1> ;     ;CALL  E_MSG               ; DISPLAY ERROR AND SET (BP) ERROR FLAG
3602                             <1> ;     ;JMP   short POD_DONE
3603                             <1>
3604                             <1> ;;HD_RESET_1:
3605                             <1> ;;    ;PUSH  BX                  ; SAVE TIMER LIMITS
3606                             <1> ;;    ;PUSH  CX
3607                             <1> ;;RES_1: MOV AH,09H              ; SET DRIVE PARAMETERS
3608                             <1> ;;    INT    13H
3609                             <1> ;;    JC     short RES_2
3610                             <1> ;;    MOV    AH,11H              ; RECALIBRATE DRIVE
3611                             <1> ;;    INT    13H
3612                             <1> ;;    JNC    short RES_CK        ; DRIVE OK
3613                             <1> ;;RES_2: ;CALL     POD_TCHK           ; CHECK TIME OUT
3614                             <1> ;;    cmp    word [wait_count], 6*182 ; waiting time (in timer ticks)
3615                             <1> ;;                                ; (30 seconds)
3616                             <1> ;;    ;cmc
3617                             <1> ;;    ;JNC   short RES_1
3618                             <1> ;;    jb     short RES_1
3619                             <1> ;;;RES_FL: ;MOV    SI,OFFSET F1781     ; INDICATE DISK 1 FAILURE;
3620                             <1> ;;    ;TEST  DL,1
3621                             <1> ;;    ;JNZ   RES_E1
3622                             <1> ;;    ;MOV   SI,OFFSET F1780     ; INDICATE DISK 0 FAILURE
3623                             <1> ;;    ;CALL  SET_FAIL            ; DO NOT TRY TO IPL DISK 0
3624                             <1> ;;    ;JMP   SHORT RES_E1
3625                             <1> ;;RES_ER: ; 22/12/2014
3626                             <1> ;;RES_OK:
3627                             <1> ;;    ;POP   CX                  ; RESTORE TIMER LIMITS
3628                             <1> ;;    ;POP   BX
3629                             <1> ;;    RETn
3630                             <1> ;;
3631                             <1> ;;RES_RS: MOV      AH,00H              ; RESET THE DRIVE
3632                             <1> ;;    INT    13H
3633                             <1> ;;RES_CK: MOV      AH,08H              ; GET MAX CYLINDER,HEAD,SECTOR
3634                             <1> ;;    MOV    BL,DL               ; SAVE DRIVE CODE
3635                             <1> ;;    INT    13H
3636                             <1> ;;    JC     short RES_ER
3637                             <1> ;;    MOV    [NEC_STATUS],CX     ; SAVE MAX CYLINDER, SECTOR
3638                             <1> ;;    MOV    DL,BL               ; RESTORE DRIVE CODE
3639                             <1> ;;RES_3: MOV AX,0401H            ; VERIFY THE LAST SECTOR
3640                             <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
00005280 80642408FE             <1>       and   byte [esp+8], 11111110b  ; clear carry bit of eflags register
                                <1>       ; 16/05/2016
00005285 1E                     <1>       push  ds
00005286 53                     <1>       push  ebx ; user's buffer address (virtual)
00005287 66BB1000               <1>       mov   bx, KDATA ; System (Kernel's) data segment
0000528B 8EDB                   <1>       mov   ds, bx
                                <1>
                                <1>       ;;15/01/2017
                                <1>       ; 14/01/2017
                                <1>       ; 02/01/2017
                                <1>       ;;mov byte [intflg], 33h  ; disk io interrupt
                                <1>       ;pop  ebx
                                <1>       ;mov  [user_buffer], ebx
                                <1>
0000528D 8F05[1C8E0100]         <1>       pop   dword [user_buffer] ; 01/06/2016
                                <1>
00005293 C605[56870100]00       <1>       mov   byte [scount], 0 ; sector count for transfer
0000529A 80FC03                 <1>       cmp   ah, 03h ; chs write
0000529D 7744                   <1>       ja    short int33h_2
```

```
3746 0000529F 7407                <1>       je    short int33h_0
3747 000052A1 80FC02              <1>       cmp   ah, 02h ; chs read
3748 000052A4 726F                <1>       jb    short int33h_5
3749 000052A6 EB68                <1>       jmp   short int33h_4
3750                              <1> int33h_0:
3751                              <1>       ; transfer user's buffer content to sector buffer
3752 000052A8 51                  <1>       push  ecx
3753 000052A9 0FB6C8              <1>       movzx ecx, al
3754                              <1> int33h_1:
3755 000052AC 56                  <1>       push  esi
3756 000052AD 8B35[1C8E0100]      <1>       mov   esi, [user_buffer]
3757                              <1>       ; esi = user's buffer address (virtual, ebx)
3758 000052B3 57                  <1>       push  edi
3759 000052B4 06                  <1>       push  es
3760 000052B5 50                  <1>       push  eax
3761 000052B6 66B81000            <1>       mov   ax, KDATA
3762 000052BA 8EC0                <1>       mov   es, ax
3763 000052BC BF00000700          <1>       mov   edi, Cluster_Buffer
3764 000052C1 C1E109              <1>       shl   ecx, 9 ; * 512
3765 000052C4 E845C00000          <1>       call  transfer_from_user_buffer
3766 000052C9 58                  <1>       pop   eax
3767 000052CA 07                  <1>       pop   es
3768 000052CB 5F                  <1>       pop   edi
3769 000052CC 5E                  <1>       pop   esi
3770 000052CD 59                  <1>       pop   ecx
3771 000052CE 7345                <1>       jnc   short int33h_5
3772 000052D0 8B1D[1C8E0100]      <1>       mov   ebx, [user_buffer] ; 01/06/2016
3773 000052D6 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 000052D7 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 000052DC B8FF000000          <1>       mov   eax, 0FFh ; Unknown error !?
3791                              <1>       ;iretd
3792 000052E1 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 000052E3 80FC05              <1>       cmp   ah, 05h ; format track
3815 000052E6 770A                <1>       ja    short int33h_3
3816 000052E8 722B                <1>       jb    short int33h_5
3817 000052EA 51                  <1>       push  ecx
3818 000052EB B901000000          <1>       mov   ecx, 1
3819 000052F0 EBBA                <1>       jmp   short int33h_1
3820                              <1> int33h_3:
3821 000052F2 80FC1C              <1>       cmp   ah, 1Ch ; LBA write
3822 000052F5 771E                <1>       ja    short int33h_5
3823 000052F7 74AF                <1>       je    short int33h_0
3824 000052F9 80FC1B              <1>       cmp   ah, 1Bh ; LBA read
3825 000052FC 7412                <1>       je    short int33h_4
3826                              <1>       ; 29/08/2020
3827 000052FE 80FC08              <1>       cmp   ah, 08h ; get disk parameters
3828                              <1>       ;jne  short int33h_5
3829 00005301 7405                <1>       je    short int33h_10
3830 00005303 80FC15              <1>       cmp   ah, 15h       ; read DASD type (get disk size)
3831 00005306 750D                <1>       jne   short int33h_5
3832                              <1> int33h_10:
3833                              <1>       ; 01/06/2016
3834 00005308 8B1D[1C8E0100]      <1>       mov   ebx, [user_buffer] ; user's buffer address
3835 0000530E EB0A                <1>       jmp   short int33h_6
3836                              <1> int33h_4:
3837 00005310 A2[56870100]        <1>       mov   byte [scount], al ; <= 128 sectors
3838                              <1> int33h_5:
3839 00005315 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 0000531A 1F                  <1>       pop   ds
3844 0000531B 9C                  <1>       pushfd
3845 0000531C 0E                  <1>       push  cs
3846 0000531D E84D000000          <1>       call  DISK_IO
3847 00005322 2E8B1D[1C8E0100]    <1>       mov   ebx, [CS:user_buffer] ; 01/06/2016
3848 00005329 723D                <1>       jc    short int33h_9
3849                              <1>       ;
3850 0000532B 2E803D[56870100]00  <1>       cmp   byte [CS:scount], 0
```

```
3851 00005333 762C                <1>       jna    short int33h_7
3852                              <1>       ;
3853                              <1>       ; transfer sector buffer content to user's buffer
3854 00005335 06                  <1>       push   es
3855 00005336 1E                  <1>       push   ds
3856 00005337 50                  <1>       push   eax
3857 00005338 66B81000            <1>       mov    ax, KDATA
3858 0000533C 8ED8                <1>       mov    ds, ax
3859 0000533E 8EC0                <1>       mov    es, ax
3860 00005340 51                  <1>       push   ecx
3861 00005341 56                  <1>       push   esi
3862 00005342 57                  <1>       push   edi
3863 00005343 0FB60D[56870100]    <1>       movzx  ecx, byte [scount]
3864 0000534A C1E109              <1>       shl    ecx, 9 ; * 512 bytes
3865 0000534D 89DF                <1>       mov    edi, ebx ; user's buffer address
3866 0000534F BE00000700          <1>       mov    esi, Cluster_Buffer
3867 00005354 E86BBF0000          <1>       call   transfer_to_user_buffer
3868 00005359 5F                  <1>       pop    edi
3869 0000535A 5E                  <1>       pop    esi
3870 0000535B 59                  <1>       pop    ecx
3871 0000535C 58                  <1>       pop    eax
3872 0000535D 1F                  <1>       pop    ds
3873 0000535E 07                  <1>       pop    es
3874 0000535F 7202                <1>       jc     short int33h_8
3875                              <1> int33h_7:
3876 00005361 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 00005362 CF                  <1>       iretd
3881                              <1> int33h_8:
3882 00005363 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 00005368 804C240801          <1>       or     byte [esp+8], 1  ; set carry bit of eflags register
3896                              <1>       ;iretd
3897 0000536D 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 0000536F 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 00005372 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 00005378 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 00005379 80FA83              <1>       CMP    DL,(80H + S_MAX_FILE - 1) ; 83h ; 30/08/2020
3921                              <1>       ;JA    short RET_2
3922 0000537C 7614                <1>       jna    short _A0
3923                              <1>       ; 29/05/2016
3924 0000537E 1E                  <1>       push   ds
3925 0000537F 50                  <1>       push   eax ; 30/08/2020 (ax --> eax)
3926 00005380 66B81000            <1>       mov    ax, KDATA
3927 00005384 8ED8                <1>       mov    ds, ax
3928 00005386 58                  <1>       pop    eax ;
3929 00005387 B4AA                <1>         mov     ah, 0AAh        ; Hard disk drive not ready !
3930                              <1>                                 ; (Programmer's guide to AMIBIOS, 1992)
3931 00005389 8825[2B820100]      <1>       mov     byte [DISK_STATUS1], ah
3932 0000538F 1F                  <1>       pop    ds
3933 00005390 EB38                <1>       jmp    short RET_2
3934                              <1> _A0:
3935                              <1>       ; 18/01/2015
3936 00005392 08E4                <1>       or     ah,ah
3937 00005394 743A                <1>       jz     short A4
3938 00005396 80FC0D              <1>       cmp    ah,0Dh ; Alternate reset
3939 00005399 7504                <1>       jne    short A2
3940 0000539B 28E4                <1>       sub    ah,ah  ; Reset
3941 0000539D EB31                <1>       jmp    short A4
3942                              <1> A2:
3943 0000539F 80FC08              <1>       CMP    AH,08H             ; GET PARAMETERS IS A SPECIAL CASE
3944                              <1>       ;JNZ   short A3
3945                              <1>         ;JMP    GET_PARM_N
3946 000053A2 0F8452030000        <1>       je     GET_PARM_N
3947 000053A8 80FC15              <1> A3:   CMP    AH,15H             ; READ DASD TYPE IS ALSO
3948                              <1>       ;JNZ   short A4
3949                              <1>         ;JMP    READ_DASD_TYPE
3950 000053AB 0F84DB020000        <1>         je      READ_DASD_TYPE
3951                              <1>       ; 02/02/2015
3952 000053B1 80FC1D              <1>       cmp    ah, 1Dh                  ;(Temporary for Retro UNIX 386 v1)
3953                              <1>       ; 12/01/2015
3954 000053B4 F5                  <1>       cmc
3955 000053B5 7319                <1>       jnc    short A4
```

```
                            <1> int33h_bad_cmd:
                            <1>       ; 16/05/2016
                            <1>       ; 30/01/2015
                            <1>       ; 29/05/2016
000053B7 1E                 <1>       push   ds
000053B8 6650               <1>       push   ax
000053BA 66B81000           <1>       mov    ax, KDATA
000053BE 8ED8               <1>       mov    ds, ax
000053C0 6658               <1>       pop    ax
000053C2 B401               <1>       mov    ah, BAD_CMD
000053C4 8825[2B820100]     <1>       mov     [DISK_STATUS1], ah ; BAD_CMD  ; COMMAND ERROR
                            <1>         ;jmp short RET_2
                            <1> RET_2:
                            <1>       ; (*) 29/05/2016
                            <1>       ; (*) retf 4
000053CA 804C240801         <1>       or     byte [esp+8], 1 ; set carry bit of eflags register
000053CF CF                 <1>       iretd
                            <1> A4:                                 ; SAVE REGISTERS DURING OPERATION
000053D0 C8080000           <1>       ENTER  8,0                    ; SAVE (BP) AND MAKE ROOM FOR @CMD_BLOCK
000053D4 53                 <1>       PUSH   eBX                    ;  IN THE STACK, THE COMMAND BLOCK IS:
000053D5 51                 <1>       PUSH   eCX                    ;   @CMD_BLOCK == BYTE PTR [BP]-8
000053D6 52                 <1>       PUSH   eDX
000053D7 1E                 <1>       PUSH   DS
000053D8 06                 <1>       PUSH   ES
000053D9 56                 <1>       PUSH   eSI
000053DA 57                 <1>       PUSH   eDI
                            <1>       ;;04/01/2015
                            <1>       ;;OR   AH,AH                  ; CHECK FOR RESET
                            <1>       ;;JNZ  short A5
                            <1>       ;;MOV  DL,80H                 ; FORCE DRIVE 80 FOR RESET
                            <1> ;;A5:
                            <1>       ;push  cs
                            <1>       ;pop   ds
                            <1>       ; 21/02/2015
000053DB 6650               <1>       push   ax
000053DD 66B81000           <1>       mov    ax, KDATA
000053E1 8ED8               <1>       mov    ds, ax
000053E3 8EC0               <1>       mov    es, ax
000053E5 6658               <1>       pop    ax
000053E7 E88D000000         <1>       CALL   DISK_IO_CONT           ; PERFORM THE OPERATION
                            <1>       ;;CALL DDS                    ; ESTABLISH SEGMENT
000053EC 8A25[2B820100]     <1>       MOV    AH,[DISK_STATUS1]      ; GET STATUS FROM OPERATION
                            <1>       ;(*) CMP AH,1                 ; SET THE CARRY FLAG TO INDICATE
                            <1>       ;(*) CMC                      ; SUCCESS OR FAILURE
000053F2 5F                 <1>       POP    eDI                    ; RESTORE REGISTERS
000053F3 5E                 <1>       POP    eSI
000053F4 07                 <1>         POP     ES
000053F5 1F                 <1>         POP     DS
000053F6 5A                 <1>       POP    eDX
000053F7 59                 <1>       POP    eCX
000053F8 5B                 <1>       POP    eBX
000053F9 C9                 <1>       LEAVE                         ; ADJUST (SP) AND RESTORE (BP)
                            <1>       ;RETf  2                      ; THROW AWAY SAVED FLAGS
                            <1>       ; (*) 29/05/2016
                            <1>       ; (*) retf 4
000053FA 80FC01             <1>       cmp    ah, 1
000053FD 7205               <1>       jc     short _A5
000053FF 804C240801         <1>       or     byte [esp+8], 1 ; set carry bit of eflags register
                            <1> _A5:
00005404 CF                 <1>       iretd
                            <1>
                            <1> ; 21/02/2015
                            <1> ;       dw --> dd
                            <1> D1:                                 ; FUNCTION TRANSFER TABLE
00005405 [C8550000]         <1>       dd     DISK_RESET             ; 000H
00005409 [3F560000]         <1>       dd     RETURN_STATUS          ; 001H
0000540D [4C560000]         <1>       dd     DISK_READ              ; 002H
00005411 [55560000]         <1>       dd     DISK_WRITE             ; 003H
00005415 [5E560000]         <1>       dd     DISK_VERF              ; 004H
00005419 [76560000]         <1>       dd     FMT_TRK                ; 005H
0000541D [BE550000]         <1>       dd     BAD_COMMAND            ; 006H FORMAT BAD SECTORS
00005421 [BE550000]         <1>       dd     BAD_COMMAND            ; 007H FORMAT DRIVE
00005425 [BE550000]         <1>       dd     BAD_COMMAND            ; 008H RETURN PARAMETERS
00005429 [93570000]         <1>       dd     INIT_DRV               ; 009H
0000542D [F2570000]         <1>       dd     RD_LONG                ; 00AH
00005431 [FB570000]         <1>       dd     WR_LONG                ; 00BH
00005435 [04580000]         <1>       dd     DISK_SEEK              ; 00CH
00005439 [C8550000]         <1>       dd     DISK_RESET             ; 00DH
0000543D [BE550000]         <1>       dd     BAD_COMMAND            ; 00EH READ BUFFER
00005441 [BE550000]         <1>       dd     BAD_COMMAND            ; 00FH WRITE BUFFER
00005445 [2C580000]         <1>       dd     TST_RDY                ; 010H
00005449 [50580000]         <1>       dd     HDISK_RECAL            ; 011H
0000544D [BE550000]         <1>       dd     BAD_COMMAND            ; 012H MEMORY DIAGNOSTIC
00005451 [BE550000]         <1>       dd     BAD_COMMAND            ; 013H DRIVE DIAGNOSTIC
00005455 [86580000]         <1>       dd     CTLR_DIAGNOSTIC        ; 014H CONTROLLER DIAGNOSTIC
                            <1>       ; 02/02/2015 (Temporary - Retro UNIX 386 v1 - DISK I/O test)
00005459 [BE550000]         <1>       dd     BAD_COMMAND            ; 015h
0000545D [BE550000]         <1>       dd     BAD_COMMAND            ; 016h
00005461 [BE550000]         <1>       dd     BAD_COMMAND            ; 017h
00005465 [BE550000]         <1>       dd     BAD_COMMAND            ; 018h
00005469 [BE550000]         <1>       dd     BAD_COMMAND            ; 019h
0000546D [BE550000]         <1>       dd     BAD_COMMAND            ; 01Ah
00005471 [4C560000]         <1>       dd     DISK_READ              ; 01Bh ; LBA read
00005475 [55560000]         <1>       dd     DISK_WRITE             ; 01Ch ; LBA write
                            <1> D1L     EQU    $ - D1
                            <1>
                            <1> DISK_IO_CONT:
                            <1>       ;;CALL DDS                    ; ESTABLISH SEGMENT
00005479 80FC01             <1>       CMP    AH,01H                 ; RETURN STATUS
                            <1>       ;;JNZ  short SU0
                            <1>         ;;JMP      RETURN_STATUS
0000547C 0F84BD010000       <1>       je     RETURN_STATUS
                            <1> SU0:
00005482 C605[2B820100]00   <1>       MOV    byte [DISK_STATUS1],0     ; RESET THE STATUS INDICATOR
                            <1>       ;;PUSH BX                     ; SAVE DATA ADDRESS
```

```
4061                                    <1>       ;mov   si, bx ;; 14/02/2015
4062 00005489 89DE                      <1>       mov    esi, ebx ; 21/02/2015
4063 0000548B 8A1D[2C820100]            <1>       MOV    BL,[HF_NUM]         ; GET NUMBER OF DRIVES
4064                                    <1>       ;; 04/01/2015
4065                                    <1>       ;;PUSH AX
4066 00005491 80E27F                    <1>       AND    DL,7FH              ; GET DRIVE AS 0 OR 1
4067                                    <1>                                  ; (get drive number as 0 to 3)
4068 00005494 38D3                      <1>       CMP    BL,DL
4069                                    <1>         ;;JBE   BAD_COMMAND_POP         ; INVALID DRIVE
4070 00005496 0F8622010000              <1>         jbe     BAD_COMMAND ;; 14/02/2015
4071                                    <1>         ;
4072                                    <1>       ;;03/01/2015
4073 0000549C 29DB                      <1>       sub    ebx, ebx
4074 0000549E 88D3                      <1>       mov    bl, dl
4075                                    <1>       ;sub   bh, bh
4076 000054A0 883D[40820100]            <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 000054A6 6650                      <1>       push   ax ; ***
4085                                    <1>       ;PUSH  ES ; **
4086 000054A8 6652                      <1>       PUSH   DX ; *
4087 000054AA 6650                      <1>       push   ax
4088 000054AC 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 000054B1 668B4310                  <1>       mov    ax, [ebx+16]
4092 000054B5 66A3[E06D0000]            <1>       mov    [HF_PORT], ax
4093                                    <1>       ;mov   dx, [ES:BX+18] ; control port address (3F6h, 376h)
4094 000054BB 668B5312                  <1>       mov    dx, [ebx+18]
4095 000054BF 668915[E26D0000]          <1>       mov    [HF_REG_PORT], dx
4096                                    <1>       ;mov   al, [ES:BX+20] ; head register upper nibble (A0h,B0h,E0h,F0h)
4097 000054C6 8A4314                    <1>       mov    al, [ebx+20]
4098                                    <1>       ; 23/02/2015
4099 000054C9 A840                      <1>       test   al, 40h       ; LBA bit (bit 6)
4100 000054CB 7406                      <1>       jz     short su1
4101 000054CD FE05[40820100]            <1>       inc    byte [LBAMode] ; 1
4102                                    <1> su1:
4103 000054D3 C0E804                    <1>       shr    al, 4
4104 000054D6 2401                      <1>       and    al, 1
4105 000054D8 A2[E46D0000]              <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 000054DD 8A4308                    <1>       mov    al, [ebx+8]
4110                                    <1>       ;MOV   DX,[HF_REG_PORT]    ; Device Control register
4111 000054E0 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 000054E1 8A25[2D820100]            <1>       MOV    AH,[CONTROL_BYTE]   ; SET EXTRA HEAD OPTION IN
4121 000054E7 80E4C0                    <1>       AND    AH,0C0H             ; CONTROL BYTE
4122 000054EA 08C4                      <1>       OR     AH,AL
4123 000054EC 8825[2D820100]            <1>       MOV    [CONTROL_BYTE],AH
4124                                    <1>       ; 04/01/2015
4125 000054F2 6658                      <1>       pop    ax
4126 000054F4 665A                      <1>       pop    dx ; * ;; 14/02/2015
4127 000054F6 20E4                      <1>       and    ah, ah ; Reset function ?
4128 000054F8 7507                      <1>       jnz    short su2
4129                                    <1>       ;;pop  dx ; * ;; 14/02/2015
4130                                    <1>       ;pop   es ; **
4131 000054FA 6658                      <1>       pop    ax ; ***
4132                                    <1>       ;;pop  bx
4133 000054FC E9C7000000                <1>         jmp     DISK_RESET
4134                                    <1> su2:
4135 00005501 803D[40820100]00          <1>       cmp    byte [LBAMode], 0
4136 00005508 7662                      <1>       jna    short su3
4137                                    <1>       ;
4138                                    <1>       ; 02/02/2015 (LBA read/write function calls)
4139 0000550A 80FC1B                    <1>       cmp    ah, 1Bh
4140 0000550D 720B                      <1>       jb     short lbarw1
4141 0000550F 80FC1C                    <1>       cmp    ah, 1Ch
4142 00005512 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 00005514 89C8                      <1>       mov    eax, ecx ; LBA address (21/02/2015)
4146                                    <1>       ;; 14/02/2015
4147 00005516 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 00005518 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 0000551A 6652                      <1>       push   dx ; * ;; 14/02/2015
4160                                    <1>       ;xor   dh, dh
4161 0000551C 31D2                      <1>       xor    edx, edx
4162                                    <1>       ;mov   dl, [ES:BX+14]      ; sectors per track (logical)
4163 0000551E 8A530E                    <1>       mov    dl, [ebx+14]
4164                                    <1>       ;xor   ah, ah
4165 00005521 31C0                      <1>       xor    eax, eax
```

```
4166                            <1>       ;mov   al, [ES:BX+2]; heads (logical)
4167 00005523 8A4302           <1>       mov    al, [ebx+2]
4168 00005526 FEC8             <1>       dec    al
4169 00005528 6640             <1>       inc    ax           ; 0 =  256
4170 0000552A 66F7E2           <1>       mul    dx
4171                            <1>              ; AX = # of Heads" * Sectors/Track
4172 0000552D 6689CA           <1>       mov    dx, cx
4173                            <1>       ;and   cx, 3Fh       ; sector  (1 to 63)
4174 00005530 83E13F           <1>       and    ecx, 3fh
4175 00005533 86D6             <1>       xchg   dl, dh
4176 00005535 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 00005538 F7E2             <1>       mul    edx
4181 0000553A 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 0000553C 01C8             <1>       add    eax, ecx
4186 0000553E 6659             <1>       pop    cx ; * ; ch = head, cl = drive number (zero based)
4187                            <1>       ;push  dx
4188                            <1>       ;push  ax
4189 00005540 50               <1>       push   eax
4190                            <1>       ;mov   al, [ES:BX+14]      ; sectors per track (logical)
4191 00005541 8A430E           <1>       mov    al, [ebx+14]
4192 00005544 F6E5             <1>       mul    ch
4193                            <1>              ;  AX = Head * Sectors/Track
4194 00005546 0FB7C0           <1>         movzx      eax, ax ; 09/12/2017
4195                            <1>       ;pop   dx
4196 00005549 5A               <1>       pop    edx
4197                            <1>       ;add   ax, dx
4198                            <1>       ;pop   dx
4199                            <1>       ;adc   dx, 0 ; add carry bit
4200 0000554A 01D0             <1>       add    eax, edx
4201                            <1> lbarw2:
4202 0000554C 29D2             <1>       sub    edx, edx ; 21/02/2015
4203 0000554E 88CA             <1>       mov    dl, cl ; 21/02/2015
4204 00005550 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 00005554 8A0D[E46D0000]   <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 0000555A 80C90E           <1>       or     cl, 0Eh ; 1110b
4218                            <1>       ;and   dh, 0Fh             ; LBA byte 4 (bits 24 to 27)
4219 0000555D 25FFFFFF0F       <1>       and    eax, 0FFFFFFFh
4220 00005562 C1E11C           <1>       shl    ecx, 28 ; 21/02/2015
4221                            <1>       ;or    dh, ch
4222 00005565 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 00005567 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 0000556A 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 0000556C 80FC14           <1>       cmp    ah, 14h
4242 0000556F 7604             <1>       jna    short chsfnc
4243                            <1> invldfnc:
4244                            <1>              ; 14/02/2015
4245                            <1>       ;pop   es ; **
4246 00005571 6658             <1>         pop     ax ; ***
4247                            <1>         ;jmp     short BAD_COMMAND_POP
4248 00005573 EB49             <1>         jmp     short BAD_COMMAND
4249                            <1> chsfnc:
4250                            <1>       ;MOV   AX,[ES:BX+5]        ; GET WRITE PRE-COMPENSATION CYLINDER
4251 00005575 668B4305         <1>       mov    ax, [ebx+5]
4252 00005579 66C1E802         <1>       SHR    AX,2
4253 0000557D 8845F8           <1>       MOV    [CMD_BLOCK],AL
4254                            <1>       ;;MOV  AL,[ES:BX+8]        ; GET CONTROL BYTE MODIFIER
4255                            <1>       ;;PUSH DX
4256                            <1>       ;;MOV  DX,[HF_REG_PORT]
4257                            <1>       ;;OUT  DX,AL               ; SET EXTRA HEAD OPTION
4258                            <1>       ;;POP  DX ; *
4259                            <1>       ;;POP  ES ; **
4260                            <1>       ;;MOV  AH,[CONTROL_BYTE]   ; SET EXTRA HEAD OPTION IN
4261                            <1>       ;;AND  AH,0C0H             ; CONTROL BYTE
4262                            <1>       ;;OR   AH,AL
4263                            <1>       ;;MOV  [CONTROL_BYTE],AH
4264                            <1>       ;
4265 00005580 88C8             <1>       MOV    AL,CL               ; GET SECTOR NUMBER
4266 00005582 243F             <1>       AND    AL,3FH
4267 00005584 8845FA           <1>       MOV    [CMD_BLOCK+2],AL
4268 00005587 886DFB           <1>       MOV    [CMD_BLOCK+3],CH    ; GET CYLINDER NUMBER
4269 0000558A 88C8             <1>       MOV    AL,CL
4270 0000558C C0E806           <1>       SHR    AL,6
```

```
4271 0000558F 8845FC              <1>       MOV    [CMD_BLOCK+4],AL    ; CYLINDER HIGH ORDER 2 BITS
4272                              <1>       ;;05/01/2015
4273                              <1>       ;;MOV  AL,DL               ; DRIVE NUMBER
4274 00005592 A0[E46D0000]        <1>       mov    al, [hf_m_s]
4275 00005597 C0E004              <1>       SHL    AL,4
4276 0000559A 80E60F              <1>       AND    DH,0FH              ; HEAD NUMBER
4277 0000559D 08F0                <1>       OR     AL,DH
4278                              <1>       ;OR    AL,80H or 20H
4279 0000559F 0CA0                <1>       OR     AL,80h+20h          ; ECC AND 512 BYTE SECTORS
4280 000055A1 8845FD              <1>       MOV    [CMD_BLOCK+5],AL    ; ECC/SIZE/DRIVE/HEAD
4281                              <1> su4:
4282                              <1>       ;POP   ES ; **
4283                              <1>         ;; 14/02/2015
4284                              <1>         ;;POP   AX
4285                              <1>         ;;MOV   [CMD_BLOCK+1],AL        ; SECTOR COUNT
4286                              <1>         ;;PUSH  AX
4287                              <1>         ;;MOV   AL,AH                   ; GET INTO LOW BYTE
4288                              <1>         ;;XOR   AH,AH                   ; ZERO HIGH BYTE
4289                              <1>         ;;SAL   AX,1                    ; *2 FOR TABLE LOOKUP
4290 000055A4 6658                <1>         pop     ax ; ***
4291 000055A6 8845F9              <1>         mov     [CMD_BLOCK+1], al
4292 000055A9 29DB                <1>         sub  ebx, ebx
4293 000055AB 88E3                <1>       mov     bl, ah
4294                              <1>         ;xor     bh, bh
4295                              <1>         ;sal     bx, 1
4296 000055AD 66C1E302            <1>         sal  bx, 2  ; 32 bit offset (21/02/2015)
4297                              <1>       ;;MOV   SI,AX                   ; PUT INTO SI FOR BRANCH
4298                              <1>         ;;CMP   AX,D1L                  ; TEST WITHIN RANGE
4299                              <1>         ;;JNB   short BAD_COMMAND_POP
4300                              <1>         ;cmp     bx, D1L
4301 000055B1 83FB74              <1>       cmp    ebx, D1L
4302 000055B4 7308                <1>       jnb    short BAD_COMMAND
4303                              <1>         ;xchg    bx, si
4304 000055B6 87DE                <1>         xchg ebx, esi
4305                              <1>       ;;;POP AX                   ; RESTORE AX
4306                              <1>       ;;;POP BX                   ; AND DATA ADDRESS
4307                              <1>
4308                              <1>       ;;PUSH CX
4309                              <1>       ;;PUSH AX                   ; ADJUST ES:BX
4310                              <1>       ;MOV   CX,BX                ; GET 3 HIGH ORDER NIBBLES OF BX
4311                              <1>       ;SHR   CX,4
4312                              <1>       ;MOV   AX,ES
4313                              <1>       ;ADD   AX,CX
4314                              <1>       ;MOV   ES,AX
4315                              <1>       ;AND   BX,000FH             ; ES:BX CHANGED TO ES:000X
4316                              <1>       ;;POP  AX
4317                              <1>       ;;POP  CX
4318                              <1>       ;;JMP  word [CS:SI+D1]
4319                              <1>       ;jmp   word [SI+D1]
4320 000055B8 FFA6[05540000]      <1>       jmp    dword [esi+D1]
4321                              <1> ;;BAD_COMMAND_POP:
4322                              <1> ;;    POP    AX
4323                              <1> ;;    POP    BX
4324                              <1> BAD_COMMAND:
4325 000055BE C605[2B820100]01    <1>         MOV     byte [DISK_STATUS1],BAD_CMD  ; COMMAND ERROR
4326 000055C5 B000                <1>       MOV    AL,0
4327 000055C7 C3                  <1>       RETn
4328                              <1>
4329                              <1> ;----------------------------------------
4330                              <1> ;     RESET THE DISK SYSTEM  (AH=00H) :
4331                              <1> ;----------------------------------------
4332                              <1>
4333                              <1> ; 18-1-2015 : one controller reset (not other one)
4334                              <1>
4335                              <1> DISK_RESET:
4336 000055C8 FA                  <1>       CLI
4337 000055C9 E4A1                <1>       IN     AL,INTB01            ; GET THE MASK REGISTER
4338                              <1>       ;JMP   $+2
4339                              <1>       IODELAY
4339 000055CB EB00                <2>  jmp short $+2
4339 000055CD EB00                <2>  jmp short $+2
4340                              <1>       ;AND   AL,0BFH              ; ENABLE FIXED DISK INTERRUPT
4341 000055CF 243F                <1>       and    al,3Fh               ; 22/12/2014 (IRQ 14 & IRQ 15)
4342 000055D1 E6A1                <1>       OUT    INTB01,AL
4343 000055D3 FB                  <1>       STI                         ; START INTERRUPTS
4344                              <1>       ; 14/02/2015
4345 000055D4 6689D7              <1>       mov    di, dx
4346                              <1>       ; 04/01/2015
4347                              <1>       ;xor   di,di
4348                              <1> drst0:
4349 000055D7 B004                <1>       MOV    AL,04H  ; bit 2 - SRST
4350                              <1>       ;MOV   DX,HF_REG_PORT
4351 000055D9 668B15[E26D0000]    <1>       MOV    DX,[HF_REG_PORT]
4352 000055E0 EE                  <1>       OUT    DX,AL                ; RESET
4353                              <1> ;     MOV    CX,10                ; DELAY COUNT
4354                              <1> ;DRD: DEC    CX
4355                              <1> ;     JNZ    short DRD            ; WAIT 4.8 MICRO-SEC
4356                              <1>       ;mov   cx,2                 ; wait for 30 micro seconds
4357 000055E1 B902000000          <1>         mov  ecx, 2 ; 21/02/2015
4358 000055E6 E851CEFFFF          <1>       call    WAITF                   ; (Award Bios 1999 - WAIT_REFRESH,
4359                              <1>                                         ; 40 micro seconds)
4360 000055EB A0[2D820100]        <1>       mov    al,[CONTROL_BYTE]
4361 000055F0 240F                <1>       AND    AL,0FH               ; SET HEAD OPTION
4362 000055F2 EE                  <1>       OUT    DX,AL                ; TURN RESET OFF
4363 000055F3 E86A040000          <1>       CALL   NOT_BUSY
4364 000055F8 7515                <1>       JNZ    short DRERR          ; TIME OUT ON RESET
4365 000055FA 668B15[E06D0000]    <1>       MOV    DX,[HF_PORT]
4366 00005601 FEC2                <1>       inc    dl  ; HF_PORT+1
4367                              <1>       ; 02/01/2015 - Award BIOS 1999 - AHDSK.ASM
4368                              <1>         ;mov     cl, 10
4369 00005603 B90A000000          <1>         mov     ecx, 10 ; 21/02/2015
4370                              <1> drst1:
4371 00005608 EC                  <1>       IN     AL,DX                ; GET RESET STATUS
4372 00005609 3C01                <1>       CMP    AL,1
4373                              <1>       ; 04/01/2015
```

```
4374 0000560B 740A                <1>       jz     short drst2
4375                              <1>       ;JNZ   short DRERR         ; BAD RESET STATUS
4376                              <1>              ; Drive/Head Register - bit 4
4377 0000560D E2F9                <1>       loop   drst1
4378                              <1> DRERR:
4379 0000560F C605[2B820100]05    <1>       MOV    byte [DISK_STATUS1],BAD_RESET ; CARD FAILED
4380 00005616 C3                  <1>       RETn
4381                              <1> drst2:
4382                              <1>       ; 14/02/2015
4383 00005617 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 0000561A A0[E46D0000]        <1>       mov    al, [hf_m_s] ; 18/01/2015
4408 0000561F A801                <1>       test   al,1
4409                              <1> ;     jnz    short drst6
4410 00005621 7516                <1>         jnz     short drst4
4411 00005623 8065FDEF            <1>       AND     byte [CMD_BLOCK+5],0EFH ; SET TO DRIVE 0
4412                              <1> ;drst5:
4413                              <1> drst3:
4414 00005627 E867010000          <1>       CALL   INIT_DRV           ; SET MAX HEADS
4415                              <1>       ;mov   dx,di
4416 0000562C 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 00005631 C605[2B820100]00    <1>       MOV    byte [DISK_STATUS1],0     ; IGNORE ANY SET UP ERRORS
4424 00005638 C3                  <1>       RETn
4425                              <1> ;drst6:
4426                              <1> drst4:            ; Drive/Head Register - bit 4
4427 00005639 804DFD10            <1>       OR      byte [CMD_BLOCK+5],010H ; SET TO DRIVE 1
4428                              <1>         ;jmp    short drst5
4429 0000563D 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 0000563F A0[2B820100]        <1>       MOV    AL,[DISK_STATUS1]   ; OBTAIN PREVIOUS STATUS
4437 00005644 C605[2B820100]00    <1>         MOV     byte [DISK_STATUS1],0   ; RESET STATUS
4438 0000564B C3                  <1>       RETn
4439                              <1>
4440                              <1> ;----------------------------------------
4441                              <1> ;     DISK READ ROUTINE    (AH = 02H) :
4442                              <1> ;----------------------------------------
4443                              <1>
4444                              <1> DISK_READ:
4445 0000564C C645FE20            <1>       MOV    byte [CMD_BLOCK+6],READ_CMD
4446 00005650 E986020000          <1>         JMP     COMMANDI
4447                              <1>
4448                              <1> ;----------------------------------------
4449                              <1> ;     DISK WRITE ROUTINE   (AH = 03H) :
4450                              <1> ;----------------------------------------
4451                              <1>
4452                              <1> DISK_WRITE:
4453 00005655 C645FE30            <1>       MOV    byte [CMD_BLOCK+6],WRITE_CMD
4454 00005659 E9D8020000          <1>         JMP     COMMANDO
4455                              <1>
4456                              <1> ;----------------------------------------
4457                              <1> ;     DISK VERIFY       (AH = 04H) :
4458                              <1> ;----------------------------------------
4459                              <1>
4460                              <1> DISK_VERF:
4461 0000565E C645FE40            <1>       MOV    byte [CMD_BLOCK+6],VERIFY_CMD
4462 00005662 E846030000          <1>       CALL   COMMAND
4463 00005667 750C                <1>       JNZ    short VERF_EXIT           ; CONTROLLER STILL BUSY
4464 00005669 E8B8030000          <1>       CALL   _WAIT               ; (Original: CALL WAIT)
4465 0000566E 7505                <1>       JNZ    short VERF_EXIT           ; TIME OUT
4466 00005670 E845040000          <1>       CALL   CHECK_STATUS
4467                              <1> VERF_EXIT:
4468 00005675 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 00005676 C645FE50            <1>       MOV    byte [CMD_BLOCK+6],FMTTRK_CMD
4476                              <1>       ;PUSH  ES
4477                              <1>       ;PUSH  BX
4478 0000567A 53                  <1>       push   ebx
```

```
4479 0000567B E8EC040000          <1>       CALL   GET_VEC             ; GET DISK PARAMETERS ADDRESS
4480                              <1>       ;MOV   AL,[ES:BX+14]       ; GET SECTORS/TRACK
4481 00005680 8A430E              <1>       mov    al, [ebx+14]
4482 00005683 8845F9              <1>       MOV    [CMD_BLOCK+1],AL    ; SET SECTOR COUNT IN COMMAND
4483 00005686 5B                  <1>       pop    ebx
4484                              <1>       ;POP   BX
4485                              <1>       ;POP   ES
4486 00005687 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 0000568C 1E                  <1>       PUSH   DS                ; SAVE REGISTERS
4497                              <1>       ;PUSH  ES
4498 0000568D 53                  <1>       PUSH   eBX
4499                              <1>       ;CALL  DDS               ; ESTABLISH ADDRESSING
4500                              <1>       ;push  cs
4501                              <1>       ;pop   ds
4502 0000568E 66BB1000            <1>         mov  bx, KDATA
4503 00005692 8EDB                <1>       mov    ds, bx
4504                              <1>       ;mov   es, bx
4505 00005694 C605[2B820100]00    <1>       MOV     byte [DISK_STATUS1],0
4506 0000569B 8A1D[2C820100]      <1>       MOV    BL,[HF_NUM]         ; GET NUMBER OF DRIVES
4507 000056A1 80E27F              <1>       AND    DL,7FH              ; GET DRIVE NUMBER
4508 000056A4 38D3                <1>       CMP    BL,DL
4509 000056A6 763C                <1>       JBE    short RDT_NOT_PRESENT     ; RETURN DRIVE NOT PRESENT
4510 000056A8 0FB6C2              <1>       movzx  eax, dl ; 28/08/2020
4511 000056AB E8BC040000          <1>       CALL   GET_VEC             ; GET DISK PARAMETERS ADDRESS
4512                              <1>
4513                              <1>       ; 28/08/2020 - TRDOS 386 v2
4514 000056B0 F6431440            <1>       test   byte [ebx+20], 40h  ; LBA bit (bit 6)
4515 000056B4 751D                <1>       jnz    short RDT3 ; LBA disk (may be > 8GB)
4516                              <1>
4517                              <1>       ;MOV   AL,[ES:BX+2]        ; HEADS
4518 000056B6 8A4302              <1>       mov    al, [ebx+2]
4519                              <1>       ;MOV   CL,[ES:BX+14]
4520 000056B9 8A4B0E              <1>       mov    cl, [ebx+14]
4521 000056BC F6E9                <1>       IMUL   CL                  ; * NUMBER OF SECTORS
4522                              <1>       ;MOV   CX,[ES:BX]          ; MAX NUMBER OF CYLINDERS
4523 000056BE 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 000056C1 6649                <1>       DEC    CX                  ; LEAVE ONE FOR DIAGNOSTICS
4529                              <1>       ;
4530 000056C3 66F7E9              <1>       IMUL   CX                  ; NUMBER OF SECTORS
4531 000056C6 6689D1              <1>       MOV    CX,DX               ; HIGH ORDER HALF
4532 000056C9 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 000056CC 29C0                <1>       sub    eax, eax ; 28/08/2020
4538                              <1>       ;
4539 000056CE 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 000056D0 5B                  <1>       POP    eBX                 ; RESTORE REGISTERS
4544                              <1>       ;POP   ES
4545 000056D1 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 000056D2 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 000056D3 C0E002             <1>       shl    al, 2 ; * 4
4560                              <1> RDT5:
4561                              <1>       ;add   eax, drv.size
4562 000056D6 05[266E0000]        <1>       add    eax, drv.size+8 ; 29/08/2020
4563 000056DB 668B10              <1>       mov    dx, [eax]   ; low word of disk size
4564 000056DE 668B4802            <1>       mov    cx, [eax+2] ; high word of disk size
4565                              <1>       ;sub   eax, eax
4566 000056E2 EBE8                <1>       jmp    short RDT4
4567                              <1>
4568                              <1> RDT_NOT_PRESENT:
4569                              <1>       ; 30/08/2020
4570 000056E4 C605[2B820100]AA    <1>       mov    byte [DISK_STATUS1], NOT_RDY ; DRIVE NOT READY
4571                              <1>
4572                              <1>       ;SUB   AX,AX               ; DRIVE NOT PRESENT RETURN
4573 000056EB 29C0                <1>       sub    eax, eax ; 30/08/2020
4574 000056ED 6689C1              <1>       MOV    CX,AX               ; ZERO BLOCK COUNT
4575 000056F0 6689C2              <1>       MOV    DX,AX
4576                              <1>       ; 30/08/2020
4577 000056F3 804C241001          <1>       or     byte [esp+16], 1 ; set carry bit of eflags register
4578                              <1>              ; cf = 1 -> ah = 0, drive not ready, disk type = 0
4579 000056F8 EBD6                <1>       JMP    short RDT2
4580                              <1>
4581                              <1> ; 28/05/2016
4582                              <1> ; 27/05/2016 - TRDOS 386 (TRDOS v2.0)
4583                              <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
000056FA 1E                 <1>       PUSH   DS                 ; SAVE REGISTERS
000056FB 06                 <1>       PUSH   ES
000056FC 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
000056FD 66BB1000           <1>       mov    bx, KDATA
00005701 8EDB               <1>       mov    ds, bx
00005703 8EC3               <1>       mov    es, bx ; 27/05/2016
                            <1>       ;
00005705 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
00005708 3A15[2C820100]     <1>       cmp    dl, [HF_NUM] ; is hard disk index < hard disk count ?
0000570E 736A               <1>       jae    short G4     ; no, error ! drive not ready !
                            <1>       ;
00005710 31DB               <1>       xor    ebx, ebx ; 21/02/2015
                            <1>       ; 22/12/2014
00005712 88D3               <1>       mov    bl, dl
                            <1>       ;xor   bh, bh
00005714 C0E302             <1>       shl    bl, 2              ; convert index to offset
                            <1>       ;add   bx, HF_TBL_VEC
00005717 81C3[30820100]     <1>       add    ebx, HF_TBL_VEC
                            <1>       ;mov   ax, [bx+2]
                            <1>       ;mov   es, ax             ; dpt segment
                            <1>       ;mov   bx, [bx]           ; dpt offset
0000571D 8B1B               <1>       mov    ebx, [ebx] ; 32 bit offset
                            <1>
0000571F C605[2B820100]00   <1>       MOV    byte [DISK_STATUS1],0
                            <1>         ;MOV AX,[ES:BX]         ; MAX NUMBER OF CYLINDERS
00005726 668B03             <1>       mov    ax, [ebx]
                            <1>       ;;SUB  AX,2               ; ADJUST FOR 0-N
00005729 6648               <1>       dec    ax                 ; max. cylinder number
0000572B 88C5               <1>       MOV    CH,AL
0000572D 66250003           <1>       AND    AX,0300H           ; HIGH TWO BITS OF CYLINDER
00005731 66D1E8             <1>       SHR    AX,1
00005734 66D1E8             <1>       SHR    AX,1
                            <1>       ;OR    AL,[ES:BX+14]      ; SECTORS
00005737 0A430E             <1>       or     al, [ebx+14]
0000573A 88C1               <1>       MOV    CL,AL
                            <1>       ;MOV   DH,[ES:BX+2]       ; HEADS
0000573C 8A7302             <1>       mov    dh, [ebx+2]
0000573F FECE               <1>       DEC    DH                 ; 0-N RANGE
00005741 8A15[2C820100]     <1>       MOV    DL,[HF_NUM]        ; DRIVE COUNT
00005747 6629C0             <1>       SUB    AX,AX
                            <1>         ;27/12/2014
                            <1>       ;mov   di, bx             ; HDPT offset
                            <1>
                            <1>       ; 29/08/2020
0000574A 833C2400           <1>       cmp    dword [esp], 0
0000574E 7703               <1>       ja     short G7 ; ebx > 0
                            <1>
                            <1>       ; if EBX (user's buffer address) = 0, do not copy DPT
00005750 5B                 <1>       pop    ebx
00005751 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>       ;
00005753 873C24             <1>       xchg   edi, [esp]         ; ebx (input)-> edi, edi -> [esp]
00005756 56                 <1>       push   esi
00005757 89DE               <1>       mov    esi, ebx           ; hard disk parameter table (32 bytes)
00005759 89FB               <1>       mov    ebx, edi           ; ebx = user's buffer address
0000575B 51                 <1>       push   ecx
0000575C 50                 <1>       push   eax
0000575D B920000000         <1>       mov    ecx, 32 ; 32 bytes
00005762 E85DBB0000         <1>       call   transfer_to_user_buffer ; trdosk6.s (16/05/2016)
00005767 58                 <1>       pop    eax
00005768 59                 <1>       pop    ecx
00005769 5E                 <1>       pop    esi
0000576A 5F                 <1>       pop    edi
0000576B 730A               <1>       jnc    short G5
                            <1>       ; 29/05/2016 (*)
0000576D B8FF000000         <1>       mov    eax, 0FFh ; unknown error !
                            <1> G6:
00005772 804C241001         <1>       or     byte [esp+16], 1 ; set carry bit of eflags register
                            <1> G5:
                            <1>       ; 27/05/2016
                            <1>       ;POP   eBX                ; RESTORE REGISTERS
00005777 07                 <1>       POP    ES
00005778 1F                 <1>       POP    DS
                            <1>       ;RETf  2
                            <1>       ; (*) 29/05/2016
                            <1>       ; (*) retf 4
                            <1>       ; (*) or byte [esp+8], 1 ; set carry bit of eflags register
00005779 CF                 <1>       iretd
                            <1> G4:
```

```
4689 0000577A C605[2B820100]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 00005781 B407                <1>       MOV    AH,INIT_FAIL
4692 00005783 28C0                <1>       SUB    AL,AL
4693                              <1>       ;SUB   DX,DX
4694                              <1>       ; 30/08/2020
4695 00005785 8A15[2C820100]      <1>       mov    dl, [HF_NUM] ; disk count
4696 0000578B 28F6                <1>       sub    dh, dh
4697 0000578D 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 00005790 5B                  <1>       pop    ebx
4703 00005791 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 00005793 31C0                <1>       xor    eax, eax ; 21/02/2015
4746 00005795 B00B                <1>       mov    al,11 ; Physical heads from translated HDPT
4747 00005797 3825[40820100]      <1>         cmp     [LBAMode], ah   ; 0
4748 0000579D 7702                <1>       ja     short idrv0
4749 0000579F B002                <1>       mov    al,2  ; Physical heads from standard HDPT
4750                              <1> idrv0:
4751                              <1>       ; DL = drive number (0 based)
4752 000057A1 E8C6030000          <1>       call   GET_VEC
4753                              <1>       ;push  bx
4754 000057A6 53                  <1>       push   ebx ; 21/02/2015
4755                              <1>       ;add   bx,ax
4756 000057A7 01C3                <1>       add    ebx, eax
4757                              <1>       ;; 05/01/2015
4758 000057A9 8A25[E46D0000]      <1>       mov    ah, [hf_m_s] ; drive number (0= master, 1= slave)
4759                              <1>       ;;and  ah,1
4760 000057AF C0E404              <1>       shl    ah,4
4761 000057B2 80CCA0              <1>       or     ah,0A0h  ; Drive/Head register - 10100000b (A0h)
4762                              <1>       ;mov   al,[es:bx]
4763 000057B5 8A03                <1>       mov    al, [ebx] ; 21/02/2015
4764 000057B7 FEC8                <1>       dec    al       ; last head number
4765                              <1>       ;and   al,0Fh
4766 000057B9 08E0                <1>       or     al,ah   ; lower 4 bits for head number
4767                              <1>       ;
4768 000057BB C645FE91            <1>       mov    byte [CMD_BLOCK+6],SET_PARM_CMD
4769 000057BF 8845FD              <1>       mov    [CMD_BLOCK+5],al
4770                              <1>       ;pop   bx
4771 000057C2 5B                  <1>       pop    ebx
4772 000057C3 29C0                <1>       sub    eax, eax ; 21/02/2015
4773 000057C5 B004                <1>       mov    al,4 ; Physical sec per track from translated HDPT
4774 000057C7 803D[40820100]00    <1>       cmp    byte [LBAMode], 0
4775 000057CE 7702                <1>       ja     short idrv1
4776 000057D0 B00E                <1>       mov    al,14 ; Physical sec per track from standard HDPT
4777                              <1> idrv1:
4778                              <1>       ;xor   ah,ah
4779                              <1>       ;add   bx,ax
4780 000057D2 01C3                <1>       add    ebx, eax ; 21/02/2015
4781                              <1>       ;mov   al,[es:bx]
4782                              <1>                    ; sector number
4783 000057D4 8A03                <1>       mov    al, [ebx]
4784 000057D6 8845F9              <1>       mov    [CMD_BLOCK+1],al
4785 000057D9 28C0                <1>       sub    al,al
4786 000057DB 8845FB              <1>       mov    [CMD_BLOCK+3],al  ; ZERO FLAGS
4787 000057DE E8CA010000          <1>       call   COMMAND        ; TELL CONTROLLER
4788 000057E3 750C                <1>       jnz    short INIT_EXIT       ; CONTROLLER BUSY ERROR
4789 000057E5 E878020000          <1>       call   NOT_BUSY       ; WAIT FOR IT TO BE DONE
4790 000057EA 7505                <1>       jnz    short INIT_EXIT       ; TIME OUT
4791 000057EC E8C9020000          <1>       call   CHECK_STATUS
4792                              <1> INIT_EXIT:
4793 000057F1 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
000057F2 C645FE22               <1>         mov     byte [CMD_BLOCK+6],READ_CMD + ECC_MODE
000057F6 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
000057FB C645FE32               <1>         MOV     byte [CMD_BLOCK+6],WRITE_CMD + ECC_MODE
000057FF E932010000             <1>         JMP     COMMANDO
                                <1>
                                <1> ;----------------------------------------
                                <1> ;     SEEK              (AH = 0CH) :
                                <1> ;----------------------------------------
                                <1>
                                <1> DISK_SEEK:
00005804 C645FE70               <1>         MOV     byte [CMD_BLOCK+6],SEEK_CMD
00005808 E8A0010000             <1>       CALL   COMMAND
0000580D 751C                   <1>       JNZ    short DS_EXIT             ; CONTROLLER BUSY ERROR
0000580F E812020000             <1>       CALL   _WAIT
00005814 7515                   <1>         JNZ     DS_EXIT                 ; TIME OUT ON SEEK
00005816 E89F020000             <1>       CALL   CHECK_STATUS
0000581B 803D[2B820100]40       <1>         CMP     byte [DISK_STATUS1],BAD_SEEK
00005822 7507                   <1>       JNE    short DS_EXIT
00005824 C605[2B820100]00       <1>         MOV     byte [DISK_STATUS1],0
                                <1> DS_EXIT:
0000582B C3                     <1>       RETn
                                <1>
                                <1> ;----------------------------------------
                                <1> ;     TEST DISK READY      (AH = 10H) :
                                <1> ;----------------------------------------
                                <1>
                                <1> TST_RDY:                          ; WAIT FOR CONTROLLER
0000582C E831020000             <1>       CALL   NOT_BUSY
00005831 751C                   <1>       JNZ    short TR_EX
00005833 8A45FD                 <1>       MOV    AL,[CMD_BLOCK+5]    ; SELECT DRIVE
00005836 668B15[E06D0000]       <1>       MOV    DX,[HF_PORT]
0000583D 80C206                 <1>       add    dl,6
00005840 EE                     <1>       OUT    DX,AL
00005841 E88C020000             <1>       CALL   CHECK_ST            ; CHECK STATUS ONLY
00005846 7507                   <1>       JNZ    short TR_EX
00005848 C605[2B820100]00       <1>       MOV    byte [DISK_STATUS1],0     ; WIPE OUT DATA CORRECTED ERROR
                                <1> TR_EX:
0000584F C3                     <1>       RETn
                                <1>
                                <1> ;----------------------------------------
                                <1> ;     RECALIBRATE       (AH = 11H) :
                                <1> ;----------------------------------------
                                <1>
                                <1> HDISK_RECAL:
00005850 C645FE10               <1>         MOV     byte [CMD_BLOCK+6],RECAL_CMD ; 10h, 16
00005854 E854010000             <1>       CALL   COMMAND             ; START THE OPERATION
00005859 7523                   <1>       JNZ    short RECAL_EXIT    ; ERROR
0000585B E8C6010000             <1>       CALL   _WAIT               ; WAIT FOR COMPLETION
00005860 7407                   <1>       JZ     short RECAL_X             ; TIME OUT ONE OK ?
00005862 E8BF010000             <1>       CALL   _WAIT               ; WAIT FOR COMPLETION LONGER
00005867 7515                   <1>       JNZ    short RECAL_EXIT    ; TIME OUT TWO TIMES IS ERROR
                                <1> RECAL_X:
00005869 E84C020000             <1>       CALL   CHECK_STATUS
0000586E 803D[2B820100]40       <1>       CMP    byte [DISK_STATUS1],BAD_SEEK ; SEEK NOT COMPLETE
00005875 7507                   <1>       JNE    short RECAL_EXIT    ; IS OK
00005877 C605[2B820100]00       <1>       MOV    byte [DISK_STATUS1],0
                                <1> RECAL_EXIT:
0000587E 803D[2B820100]00       <1>         CMP     byte [DISK_STATUS1],0
00005885 C3                     <1>       RETn
                                <1>
                                <1> ;----------------------------------------
                                <1> ;      CONTROLLER DIAGNOSTIC (AH = 14H) :
                                <1> ;----------------------------------------
                                <1>
                                <1> CTLR_DIAGNOSTIC:
00005886 FA                     <1>         CLI                             ; DISABLE INTERRUPTS WHILE CHANGING MASK
00005887 E4A1                   <1>       IN     AL,INTB01          ; TURN ON SECOND INTERRUPT CHIP
                                <1>       ;AND   AL,0BFH
00005889 243F                   <1>       and    al, 3Fh                  ; enable IRQ 14 & IRQ 15
                                <1>       ;JMP   $+2
                                <1>       IODELAY
0000588B EB00                   <2>  jmp short $+2
0000588D EB00                   <2>  jmp short $+2
0000588F E6A1                   <1>       OUT    INTB01,AL
                                <1>       IODELAY
00005891 EB00                   <2>  jmp short $+2
00005893 EB00                   <2>  jmp short $+2
00005895 E421                   <1>       IN     AL,INTA01          ; LET INTERRUPTS PASS THRU TO
00005897 24FB                   <1>       AND    AL,0FBH            ;  SECOND CHIP
                                <1>       ;JMP   $+2
                                <1>       IODELAY
00005899 EB00                   <2>  jmp short $+2
0000589B EB00                   <2>  jmp short $+2
0000589D E621                   <1>       OUT    INTA01,AL
0000589F FB                     <1>       STI
000058A0 E8BD010000             <1>       CALL   NOT_BUSY           ; WAIT FOR CARD
000058A5 752B                   <1>       JNZ    short CD_ERR       ; BAD CARD
                                <1>       ;MOV   DX, HF_PORT+7
000058A7 668B15[E06D0000]       <1>       mov    dx, [HF_PORT]
000058AE 80C207                 <1>       add    dl, 7
000058B1 B090                   <1>       MOV    AL,DIAG_CMD        ; START DIAGNOSE
```

```
4893 000058B3 EE                  <1>       OUT    DX,AL
4894 000058B4 E8A9010000          <1>       CALL   NOT_BUSY           ; WAIT FOR IT TO COMPLETE
4895 000058B9 B480                <1>       MOV    AH,TIME_OUT
4896 000058BB 7517                <1>       JNZ    short CD_EXIT            ; TIME OUT ON DIAGNOSTIC
4897                              <1>       ;MOV   DX,HF_PORT+1       ; GET ERROR REGISTER
4898 000058BD 668B15[E06D0000]    <1>       mov    dx, [HF_PORT]
4899 000058C4 FEC2                <1>       inc    dl
4900 000058C6 EC                  <1>       IN     AL,DX
4901 000058C7 A2[22820100]        <1>       MOV    [HF_ERROR],AL      ; SAVE IT
4902 000058CC B400                <1>       MOV    AH,0
4903 000058CE 3C01                <1>       CMP    AL,1               ; CHECK FOR ALL OK
4904 000058D0 7402                <1>       JE     SHORT CD_EXIT
4905 000058D2 B420                <1> CD_ERR: MOV  AH,BAD_CNTLR
4906                              <1> CD_EXIT:
4907 000058D4 8825[2B820100]      <1>       MOV    [DISK_STATUS1],AH
4908 000058DA C3                  <1>       RETn
4909                              <1>
4910                              <1> ;----------------------------------------
4911                              <1> ; COMMANDI                        :
4912                              <1> ;     REPEATEDLY INPUTS DATA TILL      :
4913                              <1> ;     NSECTOR RETURNS ZERO             :
4914                              <1> ;----------------------------------------
4915                              <1> COMMANDI:
4916 000058DB E862020000          <1>       CALL   CHECK_DMA          ; CHECK 64K BOUNDARY ERROR
4917 000058E0 7253                <1>       JC     short CMD_ABORT
4918                              <1>       ;MOV   DI,BX
4919 000058E2 89DF                <1>       mov    edi, ebx ; 21/02/2015
4920 000058E4 E8C4000000          <1>       CALL   COMMAND            ; OUTPUT COMMAND
4921 000058E9 754A                <1>       JNZ    short CMD_ABORT
4922                              <1> CMD_I1:
4923 000058EB E836010000          <1>       CALL   _WAIT              ; WAIT FOR DATA REQUEST INTERRUPT
4924 000058F0 7543                <1>       JNZ    short TM_OUT       ; TIME OUT
4925                              <1> cmd_i1x: ; 18/02/2016
4926                              <1>       ;MOV   CX,256             ; SECTOR SIZE IN WORDS
4927 000058F2 B900010000          <1>       mov    ecx, 256 ; 21/02/2015
4928                              <1>       ;MOV   DX,HF_PORT
4929 000058F7 668B15[E06D0000]    <1>       mov    dx,[HF_PORT]
4930 000058FE FA                  <1>       CLI
4931 000058FF FC                  <1>       CLD
4932 00005900 F3666D              <1>       REP    INSW               ; GET THE SECTOR
4933 00005903 FB                  <1>       STI
4934 00005904 F645FE02            <1>       TEST   byte [CMD_BLOCK+6],ECC_MODE ; CHECK FOR NORMAL INPUT
4935 00005908 7419                <1>       JZ     short CMD_I3
4936 0000590A E880010000          <1>       CALL   WAIT_DRQ           ; WAIT FOR DATA REQUEST
4937 0000590F 7224                <1>       JC     short TM_OUT
4938                              <1>       ;MOV   DX,HF_PORT
4939 00005911 668B15[E06D0000]    <1>       mov    dx,[HF_PORT]
4940                              <1>       ;MOV   CX,4               ; GET ECC BYTES
4941 00005918 B904000000          <1>       mov    ecx, 4 ; mov cx, 4
4942 0000591D EC                  <1> CMD_I2: IN   AL,DX
4943                              <1>       ;MOV   [ES:DI],AL         ; GO SLOW FOR BOARD
4944 0000591E 8807                <1>       mov    [edi], al ; 21/02/2015
4945 00005920 47                  <1>       INC    eDI
4946 00005921 E2FA                <1>       LOOP   CMD_I2
4947                              <1> CMD_I3:
4948                              <1>       ; wait for 400 ns
4949 00005923 80C207              <1>       add    dl, 7
4950 00005926 EC                  <1>       in     al, dx
4951 00005927 EC                  <1>       in     al, dx
4952 00005928 EC                  <1>       in     al, dx
4953                              <1>       ;
4954 00005929 E88C010000          <1>       CALL   CHECK_STATUS
4955 0000592E 7505                <1>       JNZ    short CMD_ABORT          ; ERROR RETURNED
4956 00005930 FE4DF9              <1>       DEC    byte [CMD_BLOCK+1]  ; CHECK FOR MORE
4957                              <1>       ;JNZ   SHORT CMD_I1
4958 00005933 75BD                <1>       jnz    short cmd_i1x ; 18/02/2016
4959                              <1> CMD_ABORT:
4960 00005935 C3                  <1> TM_OUT: RETn
4961                              <1>
4962                              <1> ;----------------------------------------
4963                              <1> ; COMMANDO                        :
4964                              <1> ;     REPEATEDLY OUTPUTS DATA TILL     :
4965                              <1> ;     NSECTOR RETURNS ZERO             :
4966                              <1> ;----------------------------------------
4967                              <1> COMMANDO:
4968 00005936 E807020000          <1>       CALL   CHECK_DMA          ; CHECK 64K BOUNDARY ERROR
4969 0000593B 72F8                <1>       JC     short CMD_ABORT
4970 0000593D 89DE                <1> CMD_OF: MOV  eSI,eBX ; 21/02/2015
4971 0000593F E869000000          <1>       CALL   COMMAND            ; OUTPUT COMMAND
4972 00005944 75EF                <1>       JNZ    short CMD_ABORT
4973 00005946 E844010000          <1>       CALL   WAIT_DRQ           ; WAIT FOR DATA REQUEST
4974 0000594B 72E8                <1>       JC     short TM_OUT              ; TOO LONG
4975                              <1> CMD_O1: ;PUSH      DS
4976                              <1>       ;PUSH  ES                 ; MOVE ES TO DS
4977                              <1>       ;POP   DS
4978                              <1>       ;MOV   CX,256             ; PUT THE DATA OUT TO THE CARD
4979                              <1>       ;MOV   DX,HF_PORT
4980                              <1>       ; 01/02/2015
4981 0000594D 668B15[E06D0000]    <1>       mov    dx, [HF_PORT]
4982                              <1>       ;push  es
4983                              <1>       ;pop   ds
4984                              <1>       ;mov   cx, 256
4985 00005954 B900010000          <1>       mov    ecx, 256 ; 21/02/2015
4986 00005959 FA                  <1>       CLI
4987 0000595A FC                  <1>       CLD
4988 0000595B F3666F              <1>       REP    OUTSW
4989 0000595E FB                  <1>       STI
4990                              <1>       ;POP   DS                 ; RESTORE DS
4991 0000595F F645FE02            <1>       TEST   byte [CMD_BLOCK+6],ECC_MODE ; CHECK FOR NORMAL OUTPUT
4992 00005963 7419                <1>       JZ     short CMD_O3
4993 00005965 E825010000          <1>       CALL   WAIT_DRQ           ; WAIT FOR DATA REQUEST
4994 0000596A 72C9                <1>       JC     short TM_OUT
4995                              <1>       ;MOV   DX,HF_PORT
4996 0000596C 668B15[E06D0000]    <1>       mov    dx, [HF_PORT]
4997                              <1>       ;MOV   CX,4               ; OUTPUT THE ECC BYTES
```

```
4998 00005973 B904000000          <1>       mov    ecx, 4  ; mov cx, 4
4999                              <1> CMD_O2: ;MOV AL,[ES:SI]
5000 00005978 8A06                <1>       mov    al, [esi]
5001 0000597A EE                  <1>       OUT    DX,AL
5002 0000597B 46                  <1>       INC    eSI
5003 0000597C E2FA                <1>       LOOP   CMD_O2
5004                              <1> CMD_O3:
5005 0000597E E8A3000000          <1>       CALL   _WAIT               ; WAIT FOR SECTOR COMPLETE INTERRUPT
5006 00005983 75B0                <1>       JNZ    short TM_OUT        ; ERROR RETURNED
5007 00005985 E830010000          <1>       CALL   CHECK_STATUS
5008 0000598A 75A9                <1>       JNZ    short CMD_ABORT
5009 0000598C F605[21820100]08    <1>       TEST   byte [HF_STATUS],ST_DRQ   ; CHECK FOR MORE
5010 00005993 75B8                <1>       JNZ    SHORT CMD_O1
5011                              <1>       ;MOV   DX,HF_PORT+2        ; CHECK RESIDUAL SECTOR COUNT
5012 00005995 668B15[E06D0000]    <1>       mov    dx, [HF_PORT]
5013                              <1>       ;add   dl, 2
5014 0000599C FEC2                <1>       inc    dl
5015 0000599E FEC2                <1>       inc    dl
5016 000059A0 EC                  <1>       IN     AL,DX               ;
5017 000059A1 A8FF                <1>       TEST   AL,0FFH             ;
5018 000059A3 7407                <1>       JZ     short CMD_O4              ; COUNT = 0  OK
5019 000059A5 C605[2B820100]BB    <1>       MOV    byte [DISK_STATUS1],UNDEF_ERR
5020                              <1>                                  ; OPERATION ABORTED - PARTIAL TRANSFER
5021                              <1> CMD_O4:
5022 000059AC 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 000059AD 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 000059AE E879FEFFFF          <1>       CALL   TST_RDY             ; CHECK DRIVE READY
5038                              <1>       ;;POP  CX
5039 000059B3 7419                <1>       JZ     short COMMAND2             ; DRIVE IS READY
5040 000059B5 803D[2B820100]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 000059BC 7507                <1>       jne    short COMMAND4
5045                              <1> CMD_TIMEOUT:
5046 000059BE C605[2B820100]20    <1>       MOV    byte [DISK_STATUS1],BAD_CNTLR
5047                              <1> COMMAND4:
5048 000059C5 5B                  <1>       POP    eBX
5049 000059C6 803D[2B820100]00    <1>         CMP     byte [DISK_STATUS1],0   ; SET CONDITION CODE FOR CALLER
5050 000059CD C3                  <1>       RETn
5051                              <1> COMMAND2:
5052 000059CE 5B                  <1>       POP    eBX
5053 000059CF 57                  <1>       PUSH   eDI
5054 000059D0 C605[23820100]00    <1>       MOV    byte [HF_INT_FLAG],0       ; RESET INTERRUPT FLAG
5055 000059D7 FA                  <1>       CLI                        ; INHIBIT INTERRUPTS WHILE CHANGING MASK
5056 000059D8 E4A1                <1>       IN     AL,INTB01           ; TURN ON SECOND INTERRUPT CHIP
5057                              <1>       ;AND   AL,0BFH
5058 000059DA 243F                <1>       and    al, 3Fh                    ; Enable IRQ 14 & 15
5059                              <1>       ;JMP   $+2
5060                              <1>       IODELAY
5060 000059DC EB00                <2>  jmp short $+2
5060 000059DE EB00                <2>  jmp short $+2
5061 000059E0 E6A1                <1>       OUT    INTB01,AL
5062 000059E2 E421                <1>       IN     AL,INTA01           ; LET INTERRUPTS PASS THRU TO
5063 000059E4 24FB                <1>       AND    AL,0FBH             ;  SECOND CHIP
5064                              <1>       ;JMP   $+2
5065                              <1>       IODELAY
5065 000059E6 EB00                <2>  jmp short $+2
5065 000059E8 EB00                <2>  jmp short $+2
5066 000059EA E621                <1>       OUT    INTA01,AL
5067 000059EC FB                  <1>       STI
5068 000059ED 31FF                <1>       XOR    eDI,eDI                    ; INDEX THE COMMAND TABLE
5069                              <1>       ;MOV   DX,HF_PORT+1        ; DISK ADDRESS
5070 000059EF 668B15[E06D0000]    <1>       mov    dx, [HF_PORT]
5071 000059F6 FEC2                <1>       inc    dl
5072 000059F8 F605[2D820100]C0    <1>       TEST   byte [CONTROL_BYTE],0C0H ; CHECK FOR RETRY SUPPRESSION
5073 000059FF 7411                <1>       JZ     short COMMAND3
5074 00005A01 8A45FE              <1>       MOV    AL, [CMD_BLOCK+6]   ; YES-GET OPERATION CODE
5075 00005A04 24F0                <1>       AND    AL,0F0H             ; GET RID OF MODIFIERS
5076 00005A06 3C20                <1>       CMP    AL,20H              ; 20H-40H IS READ, WRITE, VERIFY
5077 00005A08 7208                <1>       JB     short COMMAND3
5078 00005A0A 3C40                <1>       CMP    AL,40H
5079 00005A0C 7704                <1>       JA     short COMMAND3
5080 00005A0E 804DFE01            <1>       OR     byte [CMD_BLOCK+6],NO_RETRIES
5081                              <1>                                  ; VALID OPERATION FOR RETRY SUPPRESS
5082                              <1> COMMAND3:
5083 00005A12 8A443DF8            <1>       MOV    AL,[CMD_BLOCK+eDI]  ; GET THE COMMAND STRING BYTE
5084 00005A16 EE                  <1>       OUT    DX,AL               ; GIVE IT TO CONTROLLER
5085                              <1>       IODELAY
5085 00005A17 EB00                <2>  jmp short $+2
5085 00005A19 EB00                <2>  jmp short $+2
5086 00005A1B 47                  <1>       INC    eDI                 ; NEXT BYTE IN COMMAND BLOCK
5087 00005A1C 6642                <1>       INC    DX                  ; NEXT DISK ADAPTER REGISTER
5088 00005A1E 6683FF07            <1>       cmp    di, 7  ; 1/1/2015   ; ALL DONE?
5089 00005A22 75EE                <1>       JNZ    short COMMAND3             ; NO--GO DO NEXT ONE
5090 00005A24 5F                  <1>       POP    eDI
5091 00005A25 C3                  <1>       RETn                       ; ZERO FLAG IS SET
5092                              <1>
5093                              <1> ;CMD_TIMEOUT:
5094                              <1> ;     MOV    byte [DISK_STATUS1],BAD_CNTLR
5095                              <1> ;COMMAND4:
5096                              <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:
00005A26 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
00005A27 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
00005A2C F605[23820100]C0 <1>       test    byte [HF_INT_FLAG],0C0h
                        <1>         ;LOOPZ  WT1
00005A33 7517           <1>         JNZ     short WT3           ; INTERRUPT--LETS GO
                        <1>         ;DEC    BL
                        <1>         ;JNZ    short WT1           ; KEEP TRYING FOR A WHILE
                        <1>
                        <1> WT1_hi:
00005A35 E461           <1>         in      al, SYS1 ; 61h (PORT_B)   ; wait for lo to hi
00005A37 A810           <1>         test    al, 10h                   ; transition on memory
00005A39 75FA           <1>         jnz     short WT1_hi        ; refresh.
                        <1> WT1_lo:
00005A3B E461           <1>         in      al, SYS1            ; 061h (PORT_B)
00005A3D A810           <1>         test    al, 10h
00005A3F 74FA           <1>         jz      short WT1_lo
00005A41 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>
00005A43 C605[2B820100]80 <1> WT2:  MOV     byte [DISK_STATUS1],TIME_OUT ; REPORT TIME OUT ERROR
00005A4A EB0E           <1>         JMP     SHORT WT4
00005A4C C605[2B820100]00 <1> WT3:  MOV     byte [DISK_STATUS1],0
00005A53 C605[23820100]00 <1>       MOV     byte [HF_INT_FLAG],0
00005A5A 803D[2B820100]00 <1> WT4:  CMP     byte [DISK_STATUS1],0     ; SET CONDITION CODE FOR CALLER
00005A61 C3             <1>         RETn
                        <1>
                        <1> ;---------------------------------------
                        <1> ;       WAIT FOR CONTROLLER NOT BUSY    :
                        <1> ;---------------------------------------
                        <1> NOT_BUSY:
00005A62 FB             <1>         STI                         ; MAKE SURE INTERRUPTS ARE ON
                        <1>         ;PUSH   eBX
                        <1>         ;SUB    CX,CX               ; SET INITIAL DELAY BEFORE TEST
00005A63 668B15[E06D0000] <1>       mov     DX, [HF_PORT]
00005A6A 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
00005A6D B915160500     <1>         mov     ecx, WAIT_HDU_INT_LH  ; 21/02/2015
                        <1>         ;
                        <1> ;;      mov     byte [wait_count], 0    ; Reset wait counter
                        <1> NB1:
00005A72 EC             <1>         IN      AL,DX               ; CHECK STATUS
                        <1>         ;TEST   AL,ST_BUSY
00005A73 2480           <1>         and     al, ST_BUSY
                        <1>         ;LOOPNZ     NB1
00005A75 7410           <1>         JZ      short NB2           ; NOT BUSY--LETS GO
                        <1>         ;DEC    BL
                        <1>         ;JNZ    short NB1           ; KEEP TRYING FOR A WHILE
                        <1>
00005A77 E461           <1> NB1_hi: IN      AL,SYS1                   ; wait for hi to lo
00005A79 A810           <1>         TEST    AL,010H                   ; transition on memory
00005A7B 75FA           <1>         JNZ     SHORT NB1_hi        ; refresh.
00005A7D E461           <1> NB1_lo: IN      AL,SYS1
00005A7F A810           <1>         TEST    AL,010H
00005A81 74FA           <1>         JZ      short NB1_lo
00005A83 E2ED           <1>         LOOP    NB1
                        <1>         ;dec    bl
                        <1>         ;jnz    short NB1
                        <1>         ;
                        <1> ;;      cmp     byte [wait_count], 182  ; 10 seconds (182 timer ticks)
                        <1> ;;      jb      short NB1
                        <1>         ;
                        <1>         ;MOV    [DISK_STATUS1],TIME_OUT   ; REPORT TIME OUT ERROR
                        <1>         ;JMP    SHORT NB3
00005A85 B080           <1>         mov     al, TIME_OUT
                        <1> NB2:
                        <1>         ;MOV    byte [DISK_STATUS1],0
                        <1> ;NB3:
                        <1>         ;POP    eBX
00005A87 A2[2B820100]   <1>         mov     [DISK_STATUS1], al  ;;; will be set after return
                        <1>         ;CMP    byte [DISK_STATUS1],0     ; SET CONDITION CODE FOR CALLER
00005A8C 08C0           <1>         or      al, al              ; (zf = 0 --> timeout)
00005A8E C3             <1>         RETn
```

```
                            <1>
                            <1> ;----------------------------------------
                            <1> ;     WAIT FOR DATA REQUEST            :
                            <1> ;----------------------------------------
                            <1> WAIT_DRQ:
                            <1>       ;MOV   CX,DELAY_3
                            <1>       ;MOV   DX,HF_PORT+7
00005A8F 668B15[E06D0000]   <1>       mov    dx, [HF_PORT]
00005A96 80C207             <1>       add    dl, 7
                            <1>       ;;MOV  bl, WAIT_HDU_DRQ_HI ; 0
                            <1>       ;MOV   cx, WAIT_HDU_DRQ_LO ; 1000 (30 milli seconds)
                            <1>                                  ; (but it is written as 2000
                            <1>                                  ; micro seconds in ATORGS.ASM file
                            <1>                                  ; of Award Bios - 1999, D1A0622)
00005A99 B9E8030000         <1>       mov    ecx, WAIT_HDU_DRQ_LH ; 21/02/2015
00005A9E EC                 <1> WQ_1: IN     AL,DX               ; GET STATUS
00005A9F A808               <1>       TEST   AL,ST_DRQ           ; WAIT FOR DRQ
00005AA1 7516               <1>       JNZ    short WQ_OK
                            <1>       ;LOOP  WQ_1                ; KEEP TRYING FOR A SHORT WHILE
                            <1> WQ_hi:
00005AA3 E461               <1>       IN     AL,SYS1                   ; wait for hi to lo
00005AA5 A810               <1>       TEST   AL,010H                   ; transition on memory
00005AA7 75FA               <1>       JNZ    SHORT WQ_hi         ; refresh.
00005AA9 E461               <1> WQ_lo:  IN      AL,SYS1
00005AAB A810               <1>       TEST   AL,010H
00005AAD 74FA               <1>       JZ     SHORT WQ_lo
00005AAF E2ED               <1>       LOOP   WQ_1
                            <1>
00005AB1 C605[2B820100]80   <1>         MOV     byte [DISK_STATUS1],TIME_OUT ; ERROR
00005AB8 F9                 <1>       STC
                            <1> WQ_OK:
00005AB9 C3                 <1>       RETn
                            <1> ;WQ_OK:      ;CLC
                            <1> ;     RETn
                            <1>
                            <1> ;----------------------------------------
                            <1> ;     CHECK FIXED DISK STATUS   :
                            <1> ;----------------------------------------
                            <1> CHECK_STATUS:
00005ABA E813000000         <1>       CALL   CHECK_ST            ; CHECK THE STATUS BYTE
00005ABF 7509               <1>       JNZ    short CHECK_S1            ; AN ERROR WAS FOUND
00005AC1 A801               <1>       TEST   AL,ST_ERROR         ; WERE THERE ANY OTHER ERRORS
00005AC3 7405               <1>       JZ     short CHECK_S1            ; NO ERROR REPORTED
00005AC5 E849000000         <1>       CALL   CHECK_ER            ; ERROR REPORTED
                            <1> CHECK_S1:
00005ACA 803D[2B820100]00   <1>       CMP    byte [DISK_STATUS1],0      ; SET STATUS FOR CALLER
00005AD1 C3                 <1>       RETn
                            <1>
                            <1> ;----------------------------------------
                            <1> ;     CHECK FIXED DISK STATUS BYTE     :
                            <1> ;----------------------------------------
                            <1> CHECK_ST:
                            <1>       ;MOV   DX,HF_PORT+7        ; GET THE STATUS
00005AD2 668B15[E06D0000]   <1>       mov    dx, [HF_PORT]
00005AD9 80C207             <1>       add    dl, 7
                            <1>
                            <1>       ; 17/02/2016
                            <1>       ;(http://wiki.osdev.org/ATA_PIO_Mode)
                            <1>       ;"delay 400ns to allow drive to set new values of BSY and DRQ"
00005ADC EC                 <1>       IN     AL,DX
                            <1>       ;in    al, dx ; 100ns
                            <1>       ;in    al, dx ; 100ns
                            <1>       ;in    al, dx ; 100ns
                            <1>       NEWIODELAY ; 18/02/2016 (AWARD BIOS - 1999, 'CKST' in AHSDK.ASM)
00005ADD E6EB               <2>  out 0ebh,al
                            <1>       ;
00005ADF A2[21820100]       <1>       MOV    [HF_STATUS],AL
00005AE4 B400               <1>       MOV    AH,0
00005AE6 A880               <1>       TEST   AL,ST_BUSY          ; IF STILL BUSY
00005AE8 751A               <1>       JNZ    short CKST_EXIT           ;  REPORT OK
00005AEA B4CC               <1>       MOV    AH,WRITE_FAULT
00005AEC A820               <1>       TEST   AL,ST_WRT_FLT       ; CHECK FOR WRITE FAULT
00005AEE 7514               <1>       JNZ    short CKST_EXIT
00005AF0 B4AA               <1>       MOV    AH,NOT_RDY
00005AF2 A840               <1>       TEST   AL,ST_READY         ; CHECK FOR NOT READY
00005AF4 740E               <1>       JZ     short CKST_EXIT
00005AF6 B440               <1>       MOV    AH,BAD_SEEK
00005AF8 A810               <1>       TEST   AL,ST_SEEK_COMPL    ; CHECK FOR SEEK NOT COMPLETE
00005AFA 7408               <1>       JZ     short CKST_EXIT
00005AFC B411               <1>       MOV    AH,DATA_CORRECTED
00005AFE A804               <1>       TEST   AL,ST_CORRCTD       ; CHECK FOR CORRECTED ECC
00005B00 7502               <1>       JNZ    short CKST_EXIT
00005B02 B400               <1>       MOV    AH,0
                            <1> CKST_EXIT:
00005B04 8825[2B820100]     <1>       MOV    [DISK_STATUS1],AH   ; SET ERROR FLAG
00005B0A 80FC11             <1>       CMP    AH,DATA_CORRECTED   ; KEEP GOING WITH DATA CORRECTED
00005B0D 7403               <1>       JZ     short CKST_EX1
00005B0F 80FC00             <1>       CMP    AH,0
                            <1> CKST_EX1:
00005B12 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
00005B13 668B15[E06D0000]   <1>       mov    dx, [HF_PORT]       ;
00005B1A FEC2               <1>       inc    dl
00005B1C EC                 <1>       IN     AL,DX
00005B1D A2[22820100]       <1>       MOV    [HF_ERROR],AL
00005B22 53                 <1>       PUSH   eBX ; 21/02/2015
00005B23 B908000000         <1>       MOV    eCX,8               ; TEST ALL 8 BITS
00005B28 D0E0               <1> CK1:  SHL    AL,1                ; MOVE NEXT ERROR BIT TO CARRY
00005B2A 7202               <1>       JC     short CK2           ; FOUND THE ERROR
00005B2C E2FA               <1>       LOOP   CK1                 ; KEEP TRYING
```

```
5306 00005B2E BB[D46D0000]        <1> CK2:  MOV    eBX, ERR_TBL        ; COMPUTE ADDRESS OF
5307 00005B33 01CB                <1>       ADD    eBX,eCX                   ; ERROR CODE
5308                              <1>       ;;MOV  AH,BYTE [CS:BX]           ; GET ERROR CODE
5309                              <1>       ;mov   ah, [bx]
5310 00005B35 8A23                <1>       mov    ah, [ebx] ; 21/02/2015
5311 00005B37 8825[2B820100]      <1> CKEX: MOV    [DISK_STATUS1],AH   ; SAVE ERROR CODE
5312 00005B3D 5B                  <1>       POP    eBX
5313 00005B3E 80FC00              <1>       CMP    AH,0
5314 00005B41 C3                  <1>       RETn
5315                              <1>
5316                              <1> ;--------------------------------------------------------
5317                              <1> ; CHECK_DMA                                    :
5318                              <1> ;  -CHECK ES:BX AND # SECTORS TO MAKE SURE THAT IT WILL :
5319                              <1> ;   FIT WITHOUT SEGMENT OVERFLOW.                        :
5320                              <1> ;  -ES:BX HAS BEEN REVISED TO THE FORMAT SSSS:000X  :
5321                              <1> ;  -OK IF # SECTORS < 80H (7FH IF LONG READ OR WRITE)     :
5322                              <1> ;  -OK IF # SECTORS = 80H (7FH) AND BX <= 00H (04H) :
5323                              <1> ;  -ERROR OTHERWISE                                    :
5324                              <1> ;--------------------------------------------------------
5325                              <1> CHECK_DMA:
5326 00005B42 6650                <1>       PUSH   AX                  ; SAVE REGISTERS
5327 00005B44 66B80080            <1>       MOV    AX,8000H            ; AH = MAX # SECTORS AL = MAX OFFSET
5328 00005B48 F645FE02            <1>         TEST    byte [CMD_BLOCK+6],ECC_MODE
5329 00005B4C 7404                <1>       JZ     short CKD1
5330 00005B4E 66B8047F            <1>       MOV    AX,7F04H            ; ECC IS 4 MORE BYTES
5331 00005B52 3A65F9              <1> CKD1: CMP    AH, [CMD_BLOCK+1]   ; NUMBER OF SECTORS
5332 00005B55 7706                <1>       JA     short CKDOK         ; IT WILL FIT
5333 00005B57 7208                <1>       JB     short CKDERR        ; TOO MANY
5334 00005B59 38D8                <1>       CMP    AL,BL               ; CHECK OFFSET ON MAX SECTORS
5335 00005B5B 7204                <1>       JB     short CKDERR        ; ERROR
5336 00005B5D F8                  <1> CKDOK:       CLC                       ; CLEAR CARRY
5337 00005B5E 6658                <1>       POP    AX
5338 00005B60 C3                  <1>       RETn                       ; NORMAL RETURN
5339 00005B61 F9                  <1> CKDERR: STC                      ; INDICATE ERROR
5340 00005B62 C605[2B820100]09    <1>         MOV     byte [DISK_STATUS1],DMA_BOUNDARY
5341 00005B69 6658                <1>       POP    AX
5342 00005B6B C3                  <1>       RETn
5343                              <1>
5344                              <1> ;----------------------------------------
5345                              <1> ;     SET UP ES:BX-> DISK PARMS :
5346                              <1> ;----------------------------------------
5347                              <1>
5348                              <1> ; INPUT -> DL = 0 based drive number
5349                              <1> ; OUTPUT -> ES:BX = disk parameter table address
5350                              <1>
5351                              <1> GET_VEC:
5352                              <1>       ;SUB   AX,AX               ; GET DISK PARAMETER ADDRESS
5353                              <1>       ;MOV   ES,AX
5354                              <1>       ;TEST  DL,1
5355                              <1>       ;JZ    short GV_0
5356                              <1> ;     LES    BX,[HF1_TBL_VEC]    ; ES:BX -> DRIVE PARAMETERS
5357                              <1> ;     JMP    SHORT GV_EXIT
5358                              <1> ;GV_0:
5359                              <1> ;     LES    BX,[HF_TBL_VEC]           ; ES:BX -> DRIVE PARAMETERS
5360                              <1> ;
5361                              <1>       ;xor   bh, bh
5362 00005B6C 31DB                <1>       xor    ebx, ebx
5363 00005B6E 88D3                <1>       mov    bl, dl
5364                              <1>       ;;02/01/2015
5365                              <1>       ;;shl  bl, 1               ; port address offset
5366                              <1>       ;;mov  ax, [bx+hd_ports]   ; Base port address (1F0h, 170h)
5367                              <1>       ;;shl  bl, 1               ; dpt pointer offset
5368 00005B70 C0E302              <1>       shl    bl, 2  ;;
5369                              <1>       ;add   bx, HF_TBL_VEC            ; Disk parameter table pointer
5370 00005B73 81C3[30820100]      <1>       add    ebx, HF_TBL_VEC ; 21/02/2015
5371                              <1>       ;push  word [bx+2]         ; dpt segment
5372                              <1>       ;pop   es
5373                              <1>       ;mov   bx, [bx]            ; dpt offset
5374 00005B79 8B1B                <1>       mov    ebx, [ebx]
5375                              <1> ;GV_EXIT:
5376 00005B7B C3                  <1>       RETn
5377                              <1>
5378                              <1> hdc1_int: ; 21/02/2015
5379                              <1> ;--- HARDWARE INT 76H -- ( IRQ LEVEL  14 ) ----------------------
5380                              <1> ;                                                  :
5381                              <1> ;     FIXED DISK INTERRUPT ROUTINE                       :
5382                              <1> ;                                                  :
5383                              <1> ;--------------------------------------------------------------
5384                              <1>
5385                              <1> ; 22/12/2014
5386                              <1> ; IBM PC-XT Model 286 System BIOS Source Code - DISK.ASM (HD_INT)
5387                              <1> ;      '11/15/85'
5388                              <1> ; AWARD BIOS 1999 (D1A0622)
5389                              <1> ;     Source Code - ATORGS.ASM (INT_HDISK, INT_HDISK1)
5390                              <1>
5391                              <1> ;int_76h:
5392                              <1> HD_INT:
5393 00005B7C 6650                <1>       PUSH   AX
5394 00005B7E 1E                  <1>       PUSH   DS
5395                              <1>       ;CALL  DDS
5396                              <1>       ; 21/02/2015 (32 bit, 386 pm modification)
5397 00005B7F 66B81000            <1>       mov    ax, KDATA
5398 00005B83 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 00005B85 C605[23820100]FF    <1>       mov    byte [HF_INT_FLAG], 0FFh
5403                              <1>       ;
5404 00005B8C 6652                <1>       push   dx
5405 00005B8E 66BAF701            <1>       mov    dx, HDC1_BASEPORT+7 ; Status Register (1F7h)
5406                              <1>                                  ; Clear Controller
5407                              <1> Clear_IRQ1415:                         ; (Award BIOS - 1999)
5408 00005B92 EC                  <1>       in     al, dx              ;
5409 00005B93 665A                <1>       pop    dx
5410                              <1>       NEWIODELAY
```

```
5410 00005B95 E6EB              <2>  out 0ebh,al
5411                            <1>      ;
5412 00005B97 B020              <1>      MOV    AL,EOI              ; NON-SPECIFIC END OF INTERRUPT
5413 00005B99 E6A0              <1>      OUT    INTB00,AL           ; FOR CONTROLLER #2
5414                            <1>      ;JMP   $+2                 ; WAIT
5415                            <1>      NEWIODELAY
5415 00005B9B E6EB              <2>  out 0ebh,al
5416 00005B9D E620              <1>      OUT    INTA00,AL           ; FOR CONTROLLER #1
5417 00005B9F 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 00005BA0 6658              <1>      POP    AX
5423 00005BA2 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 00005BA3 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 00005BA5 B00B              <1>      mov    al, 0Bh  ; In-Service Register
5438 00005BA7 E6A0              <1>      out    0A0h, al
5439 00005BA9 EB00              <1>        jmp short $+2
5440 00005BAB EB00              <1>      jmp short $+2
5441 00005BAD E4A0              <1>      in     al, 0A0h
5442 00005BAF 2480              <1>      and    al, 80h ; bit 7 (is it real IRQ 15 or fake?)
5443 00005BB1 74ED              <1>      jz     short irq15_iret ; Fake (spurious)IRQ, do not send EOI)
5444                            <1>      ;
5445 00005BB3 1E                <1>      PUSH   DS
5446                            <1>      ;CALL  DDS
5447                            <1>      ; 21/02/2015 (32 bit, 386 pm modification)
5448 00005BB4 66B81000          <1>      mov    ax, KDATA
5449 00005BB8 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 00005BBA 800D[23820100]C0  <1>      or     byte [HF_INT_FLAG], 0C0h
5454                            <1>      ;
5455 00005BC1 6652              <1>      push   dx
5456 00005BC3 66BA7701          <1>      mov    dx, HDC2_BASEPORT+7 ; Status Register (177h)
5457                            <1>                                 ; Clear Controller (Award BIOS 1999)
5458 00005BC7 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 ///
2887                                %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
```

```
<1> SWP_DISK_READ_ERR     equ 40
<1> SWP_DISK_NOT_PRESENT_ERR   equ 41
<1> SWP_SECTOR_NOT_PRESENT_ERR equ 42
<1> SWP_NO_FREE_SPACE_ERR      equ 43
<1> SWP_DISK_WRITE_ERR         equ 44
<1> SWP_NO_PAGE_TO_SWAP_ERR    equ 45
<1> PTE_A_ACCESS_BIT equ 5  ; Bit 5 (accessed flag)
<1> SECTOR_SHIFT     equ 3  ; sector shift (to convert page block number)
<1> ; 12/07/2016
<1> PTE_SHARED   equ 400h           ; AVL bit 1, direct memory access bit
<1>                                 ; (Indicates that the page is not allocated
<1>                                 ; for the process, it is a shared or system
<1>                                         ; page, it must not be deallocated!)
<1> ; 14/12/2020
<1> ; (Linear Frame Buffer - video memory mark : AVL bit 1, outside M.A.T.)
<1> PDE_EXTERNAL equ 400h     ; Page directory entry for external memory blocks
<1> PTE_EXTERNAL equ 400h     ; Allocated kernel pages for Linear Frame Buffer
<1>                           ; (Out of memory allocation table)
<1>
<1> ;
<1> ;; Retro Unix 386 v1 - paging method/principles
<1> ;;
<1> ;; 10/10/2014
<1> ;; RETRO UNIX 386 v1 - PAGING METHOD/PRINCIPLES
<1> ;;
<1> ;; KERNEL PAGE MAP: 1 to 1 physical memory page map
<1> ;;    (virtual address = physical address)
<1> ;; KERNEL PAGE TABLES:
<1> ;;    Kernel page directory and all page tables are
<1> ;;    on memory as initialized, as equal to physical memory
<1> ;;    layout. Kernel pages can/must not be swapped out/in.
<1> ;;
<1> ;;    what for: User pages may be swapped out, when accessing
<1> ;;    a page in kernel/system mode, if it would be swapped out,
<1> ;;    kernel would have to swap it in! But it is also may be
<1> ;;    in use by a user process. (In system/kernel mode
<1> ;;    kernel can access all memory pages even if they are
<1> ;;    reserved/allocated for user processes. Swap out/in would
<1> ;;    cause conflicts.)
<1> ;;
<1> ;;    As result of these conditions,
<1> ;;    all kernel pages must be initialized as equal to
<1> ;;    physical layout for preventing page faults.
<1> ;;    Also, calling "allocate page" procedure after
<1> ;;    a page fault can cause another page fault (double fault)
<1> ;;    if all kernel page tables would not be initialized.
<1> ;;
<1> ;;    [first_page] = Beginning of users space, as offset to
<1> ;;    memory allocation table. (double word aligned)
<1> ;;
<1> ;;    [next_page] = first/next free space to be searched
<1> ;;    as offset to memory allocation table. (dw aligned)
<1> ;;
<1> ;;    [last_page] = End of memory (users space), as offset
<1> ;;    to memory allocation table. (double word aligned)
<1> ;;
<1> ;; USER PAGE TABLES:
<1> ;;    Demand paging (& 'copy on write' allocation method) ...
<1> ;;           'ready only' marked copies of the
<1> ;;           parent process's page table entries (for
<1> ;;           same physical memory).
<1> ;;           (A page will be copied to a new page after
<1> ;;            if it causes R/W page fault.)
<1> ;;
<1> ;;    Every user process has own (different)
<1> ;;    page directory and page tables.
<1> ;;
<1> ;;    Code starts at virtual address 0, always.
<1> ;;    (Initial value of EIP is 0 in user mode.)
<1> ;;    (Programs can be written/developed as simple
<1> ;;     flat memory programs.)
<1> ;;
<1> ;; MEMORY ALLOCATION STRATEGY:
<1> ;;    Memory page will be allocated by kernel only
<1> ;;          (in kernel/system mode only).
<1> ;;    * After a
<1> ;;      - 'not present' page fault
<1> ;;      - 'writing attempt on read only page' page fault
<1> ;;    * For loading (opening, reading) a file or disk/drive
<1> ;;    * As responce to 'allocate additional memory blocks'
<1> ;;      request by running process.
<1> ;;    * While creating a process, allocating a new buffer,
<1> ;;      new page tables etc.
<1> ;;
<1> ;;    At first,
<1> ;;    - 'allocate page' procedure will be called;
<1> ;,       if it will return with a valid (>0) physical address
<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>       ;
 00005BC9 A1[98810100]  <1>       mov   eax, [free_pages]
 00005BCE 21C0          <1>       and   eax, eax
 00005BD0 7438          <1>       jz    short out_of_memory
                        <1>       ;
 00005BD2 53            <1>       push  ebx
 00005BD3 51            <1>       push  ecx
                        <1>       ;
 00005BD4 BB00001000    <1>       mov   ebx, MEM_ALLOC_TBL   ; Memory Allocation Table offset
 00005BD9 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)
 00005BDB 031D[9C810100]<1>       add   ebx, [next_page] ; Free page searching starts from here
                        <1>                              ; next_free_page >> 5
 00005BE1 030D[A0810100]<1>       add   ecx, [last_page] ; Free page searching ends here
                        <1>                              ; (total_pages - 1) >> 5
                        <1> al_p_scan:
 00005BE7 39CB          <1>       cmp   ebx, ecx
 00005BE9 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>       ;
00005BEB 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.
00005BEE 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!)
00005BF0 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.
00005BF3 EBF2           <1>         jmp     short al_p_scan
                        <1>       ;
                        <1> al_p_notfound:
00005BF5 81E900001000   <1>       sub    ecx, MEM_ALLOC_TBL
00005BFB 890D[9C810100] <1>       mov    [next_page], ecx ; next/first free page = last page
                        <1>                               ; (deallocate_page procedure will change it)
00005C01 31C0           <1>       xor    eax, eax
00005C03 A3[98810100]   <1>       mov    [free_pages], eax ; 0
00005C08 59             <1>       pop    ecx
00005C09 5B             <1>       pop    ebx
                        <1>       ;
                        <1> out_of_memory:
00005C0A E85B040000     <1>       call   swap_out
00005C0F 7325           <1>       jnc    short al_p_ok  ; [free_pages] = 0, re-allocation by swap_out
                        <1>       ;
00005C11 29C0           <1>       sub    eax, eax ; 0
00005C13 F9             <1>       stc
00005C14 C3             <1>       retn
                        <1>
                        <1> al_p_found:
00005C15 89D9           <1>       mov    ecx, ebx
00005C17 81E900001000   <1>       sub    ecx, MEM_ALLOC_TBL
00005C1D 890D[9C810100] <1>       mov    [next_page], ecx ; Set first free page searching start
                        <1>                               ; address/offset (to the next)
00005C23 FF0D[98810100] <1>         dec     dword [free_pages] ; 1 page has been allocated (X = X-1)
                        <1>       ;
00005C29 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)
00005C2C C1E103         <1>       shl    ecx, 3        ; (page block offset * 32) + page index
00005C2F 01C8           <1>       add    eax, ecx      ; = page number
00005C31 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...
00005C34 59             <1>       pop    ecx
00005C35 5B             <1>       pop    ebx
                        <1> al_p_ok:
00005C36 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>       ;
00005C37 E88DFFFFFF     <1>       call   allocate_page
00005C3C 7216           <1>       jc     short mkpd_error
                        <1>       ;
00005C3E 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>       ;
00005C43 57                     <1>       push   edi
00005C44 51                     <1>       push   ecx
00005C45 50                     <1>       push   eax
00005C46 B900040000             <1>       mov    ecx, PAGE_SIZE / 4
00005C4B 89C7                   <1>       mov    edi, eax
00005C4D 31C0                   <1>       xor    eax, eax
00005C4F F3AB                   <1>       rep    stosd
00005C51 58                     <1>       pop    eax
00005C52 59                     <1>       pop    ecx
00005C53 5F                     <1>       pop    edi
                                <1> mkpd_error:
                                <1> mkpt_error:
00005C54 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>       ;
00005C55 E86FFFFFFF             <1>       call   allocate_page
00005C5A 72F8                   <1>       jc     short mkpt_error
00005C5C E811000000             <1>       call   set_pde
00005C61 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>       ;
00005C63 E861FFFFFF             <1>       call   allocate_page
00005C68 7207                   <1>       jc     short mkp_err
00005C6A E821000000             <1>       call   set_pte
00005C6F 73D2                   <1>       jnc    short clear_page ; 18/04/2015
                                <1> mkp_err:
00005C71 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>       ;
00005C72 89DA                   <1>       mov    edx, ebx
00005C74 C1EA16                 <1>       shr    edx, PAGE_D_SHIFT ; 22
00005C77 C1E202                 <1>       shl    edx, 2 ; offset to page directory (1024*4)
00005C7A 0315[B8030300]         <1>       add    edx, [u.pgdir]
                                <1>       ;
00005C80 21C0                   <1>       and    eax, eax
00005C82 7506                   <1>       jnz    short spde_1
                                <1>       ;
00005C84 8B02                   <1>       mov    eax, [edx]  ; old PDE value
                                <1>       ;test  al, 1
                                <1>       ;jz    short spde_2
00005C86 662500F0               <1>       and    ax, PDE_A_CLEAR ; 0F000h  ; clear lower 12 bits
                                <1> spde_1:
                                <1>       ;and   cx, 0FFFh
```

```
572 00005C8A 8902              <1>       mov   [edx], eax
573 00005C8C 66090A            <1>       or    [edx], cx
574 00005C8F C3                <1>       retn
575                            <1> ;spde_2: ; error
576                            <1> ;     stc
577                            <1> ;     retn
578                            <1>
579                            <1> set_pte:    ; Set page table entry (PTE)
580                            <1>       ; 24/07/2015
581                            <1>       ; 20/07/2015
582                            <1>       ; 23/06/2015
583                            <1>       ; 18/04/2015
584                            <1>       ; 12/04/2015
585                            <1>       ; 10/10/2014 ; (Retro UNIX 386 v1 - beginning)
586                            <1>       ;
587                            <1>       ; INPUT ->
588                            <1>       ;     EAX = physical page address
589                            <1>       ;           (use present value if EAX = 0)
590                            <1>       ;     EBX = virtual (linear) address
591                            <1>       ;     ECX = page attributes (lower 12 bits)
592                            <1>       ;           (higher 20 bits must be ZERO)
593                            <1>       ;           (bit 0 must be 1)
594                            <1>       ;     u.pgdir = page directory (physical) address
595                            <1>       ; OUTPUT ->
596                            <1>       ;     EAX = physical page address
597                            <1>       ;     (EDX = PTE value)
598                            <1>       ;     EBX = virtual address
599                            <1>       ;
600                            <1>       ;     CF = 1 -> error
601                            <1>       ;
602                            <1>       ; Modified Registers -> EAX, EDX
603                            <1>       ;
604 00005C90 50               <1>       push  eax
605 00005C91 A1[B8030300]      <1>       mov   eax, [u.pgdir] ; 20/07/2015
606 00005C96 E837000000        <1>       call  get_pde
607                            <1>             ; EDX = PDE address
608                            <1>             ; EAX = PDE value
609 00005C9B 5A                <1>       pop   edx ; physical page address
610 00005C9C 722A              <1>       jc    short spte_err ; PDE not present
611                            <1>       ;
612 00005C9E 53                <1>       push  ebx ; 24/07/2015
613 00005C9F 662500F0          <1>       and   ax, PDE_A_CLEAR ; 0F000h ; clear lower 12 bits
614                            <1>                       ; EDX = PT address (physical)
615 00005CA3 C1EB0C            <1>       shr   ebx, PAGE_SHIFT ; 12
616 00005CA6 81E3FF030000      <1>       and   ebx, PTE_MASK; 03FFh
617                            <1>                     ; clear higher 10 bits (PD bits)
618 00005CAC C1E302            <1>       shl   ebx, 2   ; offset to page table (1024*4)
619 00005CAF 01C3              <1>       add   ebx, eax
620                            <1>       ;
621 00005CB1 8B03              <1>       mov   eax, [ebx] ; Old PTE value
622 00005CB3 A801              <1>       test  al, 1
623 00005CB5 740C              <1>       jz    short spte_0
624 00005CB7 09D2              <1>       or    edx, edx
625 00005CB9 750F              <1>       jnz   short spte_1
626 00005CBB 662500F0          <1>       and   ax, PTE_A_CLEAR ; 0F000h ; clear lower 12 bits
627 00005CBF 89C2              <1>       mov   edx, eax
628 00005CC1 EB09              <1>       jmp   short spte_2
629                            <1> spte_0:
630                            <1>       ; If this PTE contains a swap (disk) address,
631                            <1>       ; it can be updated by using 'swap_in' procedure
632                            <1>       ; only!
633 00005CC3 21C0              <1>       and   eax, eax
634 00005CC5 7403              <1>       jz    short spte_1
635                            <1>       ; 24/07/2015
636                            <1>       ; swapped page ! (on disk)
637 00005CC7 5B                <1>       pop   ebx
638                            <1> spte_err:
639 00005CC8 F9                <1>       stc
640 00005CC9 C3                <1>       retn
641                            <1> spte_1:
642 00005CCA 89D0              <1>       mov   eax, edx
643                            <1> spte_2:
644 00005CCC 09CA              <1>       or    edx, ecx
645                            <1>       ; 23/06/2015
646 00005CCE 8913              <1>       mov   [ebx], edx ; PTE value in EDX
647                            <1>       ; 24/07/2015
648 00005CD0 5B                <1>       pop   ebx
649 00005CD1 C3                <1>       retn
650                            <1>
651                            <1> get_pde:    ; Get present value of the relevant PDE
652                            <1>       ; 20/07/2015
653                            <1>       ; 18/04/2015
654                            <1>       ; 12/04/2015
655                            <1>       ; 10/10/2014 ; (Retro UNIX 386 v1 - beginning)
656                            <1>       ;
657                            <1>       ; INPUT ->
658                            <1>       ;     EBX = virtual (linear) address
659                            <1>       ;     EAX = page directory (physical) address
660                            <1>       ; OUTPUT ->
661                            <1>       ;     EDX = Page directory entry address
662                            <1>       ;     EAX = Page directory entry value
663                            <1>       ;     CF = 1 -> PDE not present or invalid ?
664                            <1>       ; Modified Registers -> EDX, EAX
665                            <1>       ;
666 00005CD2 89DA              <1>       mov   edx, ebx
667 00005CD4 C1EA16            <1>       shr   edx, PAGE_D_SHIFT ; 22  (12+10)
668 00005CD7 C1E202            <1>       shl   edx, 2 ; offset to page directory (1024*4)
669 00005CDA 01C2              <1>       add   edx, eax ; page directory address (physical)
670 00005CDC 8B02              <1>       mov   eax, [edx]
671 00005CDE A801              <1>       test  al, PDE_A_PRESENT ; page table is present or not !
672 00005CE0 751F              <1>       jnz   short gpte_retn
673 00005CE2 F9                <1>       stc
674                            <1> gpde_retn:
675 00005CE3 C3                <1>       retn
676                            <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>       ;
00005CE4 E8E9FFFFFF     <1>       call  get_pde
00005CE9 72F8           <1>       jc    short gpde_retn     ; page table is not present
                        <1>       ;jnc  short gpte_1
                        <1>       ;retn
                        <1> ;gpte_1:
00005CEB 662500F0       <1>       and   ax, PDE_A_CLEAR ; 0F000h ; clear lower 12 bits
00005CEF 89DA           <1>       mov   edx, ebx
00005CF1 C1EA0C         <1>       shr   edx, PAGE_SHIFT ; 12
00005CF4 81E2FF030000   <1>       and   edx, PTE_MASK; 03FFh
                        <1>                  ; clear higher 10 bits (PD bits)
00005CFA C1E202         <1>       shl   edx, 2 ; offset from start of page table (1024*4)
00005CFD 01C2           <1>       add   edx, eax
00005CFF 8B02           <1>       mov   eax, [edx]
                        <1> gpte_retn:
00005D01 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>       ;
00005D02 56             <1>       push  esi
00005D03 51             <1>       push  ecx
00005D04 50             <1>       push  eax
00005D05 89C6           <1>       mov   esi, eax
00005D07 31C9           <1>       xor   ecx, ecx
                        <1>       ; The 1st PDE points to Kernel Page Table 0 (the 1st 4MB),
                        <1>       ; it must not be deallocated
00005D09 890E           <1>       mov   [esi], ecx ; 0 ; clear PDE 0
                        <1> dapd_0:
00005D0B AD             <1>       lodsd
00005D0C A801           <1>       test  al, PDE_A_PRESENT ; bit 0, present flag (must be 1)
00005D0E 7409           <1>       jz    short dapd_1
00005D10 662500F0       <1>       and   ax, PDE_A_CLEAR ; 0F000h ; clear lower 12 (attribute) bits
00005D14 E812000000     <1>       call  deallocate_page_table
                        <1> dapd_1:
00005D19 41             <1>       inc   ecx ; page directory entry index
00005D1A 81F900040000   <1>       cmp   ecx, PAGE_SIZE / 4 ; 1024
00005D20 72E9           <1>       jb    short dapd_0
                        <1> dapd_2:
00005D22 58             <1>       pop   eax
00005D23 E87F000000     <1>       call  deallocate_page     ; deallocate the page dir's itself
00005D28 59             <1>       pop   ecx
00005D29 5E             <1>       pop   esi
00005D2A 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>       ;
00005D2B 56             <1>       push  esi
00005D2C 57             <1>       push  edi
00005D2D 52             <1>       push  edx
```

```
00005D2E 50                <1>       push   eax ; *
00005D2F 89C6              <1>       mov    esi, eax
00005D31 31FF              <1>       xor    edi, edi ; 0
                           <1> dapt_0:
00005D33 AD                <1>       lodsd
00005D34 A801              <1>       test   al, PTE_A_PRESENT ; bit 0, present flag (must be 1)
00005D36 7441              <1>       jz     short dapt_1
                           <1>       ;
00005D38 A802              <1>       test   al, PTE_A_WRITE   ; bit 1, writable (r/w) flag
                           <1>                                ; (must be 1)
00005D3A 754C              <1>       jnz    short dapt_3
                           <1>       ; Read only -duplicated- page (belongs to a parent or a child)
00005D3C 66A90002          <1>         test    ax, PTE_DUPLICATED ; Was this page duplicated
                           <1>                                    ; as child's page ?
00005D40 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)
00005D42 53                <1>       push   ebx
00005D43 51                <1>       push   ecx
00005D44 66C1E102          <1>       shl    cx, 2 ; *4
00005D48 01CB              <1>       add    ebx, ecx ; PDE offset (for the parent)
00005D4A 8B0B              <1>       mov    ecx, [ebx]
00005D4C F6C101            <1>       test   cl, PDE_A_PRESENT ; present (valid) or not ?
00005D4F 7435              <1>       jz     short dapt_2 ; parent process does not use this page
00005D51 6681E100F0        <1>       and    cx, PDE_A_CLEAR ; 0F000h ; Clear attribute bits
                           <1>       ; EDI = page table entry index (0-1023)
00005D56 89FA              <1>       mov    edx, edi
00005D58 66C1E202          <1>       shl    dx, 2 ; *4
00005D5C 01CA              <1>       add    edx, ecx ; PTE offset (for the parent)
00005D5E 8B1A              <1>       mov    ebx, [edx]
00005D60 F6C301            <1>       test   bl, PTE_A_PRESENT ; present or not ?
00005D63 7421              <1>       jz     short dapt_2 ; parent process does not use this page
00005D65 662500F0          <1>       and    ax, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
00005D69 6681E300F0        <1>       and    bx, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
00005D6E 39D8              <1>       cmp    eax, ebx     ; parent's and child's pages are same ?
00005D70 7514              <1>       jne    short dapt_2 ; not same page
                           <1>                           ; deallocate the child's page
00005D72 800A02            <1>         or     byte [edx], PTE_A_WRITE ; convert to writable page (parent)
00005D75 59                <1>       pop    ecx
00005D76 5B                <1>       pop    ebx
00005D77 EB1A              <1>       jmp    short dapt_4
                           <1> dapt_1:
00005D79 09C0              <1>       or     eax, eax     ; swapped page ?
00005D7B 741D              <1>       jz     short dapt_5 ; no
                           <1>                           ; yes
00005D7D D1E8              <1>       shr    eax, 1
00005D7F E8CA040000        <1>       call   unlink_swap_block ; Deallocate swapped page block
                           <1>                                ; on the swap disk (or in file)
00005D84 EB14              <1>       jmp    short dapt_5
                           <1> dapt_2:
00005D86 59                <1>       pop    ecx
00005D87 5B                <1>       pop    ebx
                           <1> dapt_3:
                           <1>       ; 12/07/2016
00005D88 66A90004          <1>       test   ax, PTE_SHARED ; shared or direct memory access indicator
00005D8C 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
00005D8E E814000000        <1>       call   deallocate_page ; set the mem allocation bit of this page
                           <1> dapt_4:
00005D93 C746FC00000000    <1>       mov    dword [esi-4], 0 ; clear/reset PTE (child, dupl. as parent)
                           <1> dapt_5:
00005D9A 47                <1>       inc    edi ; page table entry index
00005D9B 81FF00040000      <1>       cmp    edi, PAGE_SIZE / 4 ; 1024
00005DA1 7290              <1>       jb     short dapt_0
                           <1>       ;
00005DA3 58                <1>       pop    eax ; *
00005DA4 5A                <1>       pop    edx
00005DA5 5F                <1>       pop    edi
00005DA6 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>       ;
00005DA7 53                <1>       push   ebx
00005DA8 52                <1>       push   edx
                           <1>       ;
00005DA9 C1E80C            <1>       shr    eax, PAGE_SHIFT      ; shift physical address to
                           <1>                                   ; 12 bits right
                           <1>                                   ; to get page number
00005DAC 89C2              <1>       mov    edx, eax
                           <1>       ; 15/09/2015
00005DAE C1EA03            <1>       shr    edx, 3            ; to get offset to M.A.T.
                           <1>                                  ; (1 allocation bit = 1 page)
                           <1>                                  ; (1 allocation bytes = 8 pages)
00005DB1 80E2FC            <1>       and    dl, 0FCh          ; clear lower 2 bits
                           <1>                                  ; (to get 32 bit position)
                           <1>       ;
```

```
887 00005DB4 BB00001000          <1>       mov    ebx, MEM_ALLOC_TBL   ; Memory Allocation Table address
888 00005DB9 01D3                <1>       add    ebx, edx
889 00005DBB 83E01F              <1>       and    eax, 1Fh          ; lower 5 bits only
890                              <1>                                ; (allocation bit position)
891 00005DBE 3B15[9C810100]      <1>       cmp    edx, [next_page]     ; is the new free page address lower
892                              <1>                                ; than the address in 'next_page' ?
893                              <1>                                ; (next/first free page value)
894 00005DC4 7306                <1>       jnb    short dap_1       ; no
895 00005DC6 8915[9C810100]      <1>       mov    [next_page], edx     ; yes
896                              <1> dap_1:
897 00005DCC 0FAB03              <1>       bts    [ebx], eax        ; unlink/release/deallocate page
898                              <1>                                ; set relevant bit to 1.
899                              <1>                                ; set CF to the previous bit value
900                              <1>       ;cmc                     ; complement carry flag
901                              <1>       ;jc    short dap_2       ; do not increase free_pages count
902                              <1>                                ; if the page is already deallocated
903                              <1>                                ; before.
904 00005DCF FF05[98810100]      <1>         inc     dword [free_pages]
905                              <1> dap_2:
906 00005DD5 5A                  <1>       pop    edx
907 00005DD6 5B                  <1>       pop    ebx
908 00005DD7 C3                  <1>       retn
909                              <1>
910                              <1> ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
911                              <1> ;;                                                                    ;;
912                              <1> ;; Copyright (C) KolibriOS team 2004-2012. All rights reserved.       ;;
913                              <1> ;; Distributed under terms of the GNU General Public License          ;;
914                              <1> ;;                                                                    ;;
915                              <1> ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
916                              <1>
917                              <1> ;;$Revision: 5057 $
918                              <1>
919                              <1>
920                              <1> ;;align 4
921                              <1> ;;proc alloc_page
922                              <1>
923                              <1> ;;        pushfd
924                              <1> ;;        cli
925                              <1> ;;        push    ebx
926                              <1> ;;;//-
927                              <1> ;;        cmp     [pg_data.pages_free], 1
928                              <1> ;;        jle     .out_of_memory
929                              <1> ;;;//-
930                              <1> ;;
931                              <1> ;;        mov     ebx, [page_start]
932                              <1> ;;        mov     ecx, [page_end]
933                              <1> ;;.l1:
934                              <1> ;;        bsf     eax, [ebx];
935                              <1> ;;        jnz     .found
936                              <1> ;;        add     ebx, 4
937                              <1> ;;        cmp     ebx, ecx
938                              <1> ;;        jb      .l1
939                              <1> ;;        pop     ebx
940                              <1> ;;        popfd
941                              <1> ;;        xor     eax, eax
942                              <1> ;;        ret
943                              <1> ;;.found:
944                              <1> ;;;//-
945                              <1> ;;        dec     [pg_data.pages_free]
946                              <1> ;;        jz      .out_of_memory
947                              <1> ;;;//-
948                              <1> ;;        btr     [ebx], eax
949                              <1> ;;        mov     [page_start], ebx
950                              <1> ;;        sub     ebx, sys_pgmap
951                              <1> ;;        lea     eax, [eax+ebx*8]
952                              <1> ;;        shl     eax, 12
953                              <1> ;;;//-       dec [pg_data.pages_free]
954                              <1> ;;        pop     ebx
955                              <1> ;;        popfd
956                              <1> ;;        ret
957                              <1> ;;;//-
958                              <1> ;;.out_of_memory:
959                              <1> ;;        mov     [pg_data.pages_free], 1
960                              <1> ;;        xor     eax, eax
961                              <1> ;;        pop     ebx
962                              <1> ;;        popfd
963                              <1> ;;        ret
964                              <1> ;;;//-
965                              <1> ;;endp
966                              <1>
967                              <1> duplicate_page_dir:
968                              <1>       ; 21/09/2015
969                              <1>       ; 31/08/2015
970                              <1>       ; 20/07/2015
971                              <1>       ; 28/04/2015
972                              <1>       ; 27/04/2015
973                              <1>       ; 18/04/2015
974                              <1>       ; 12/04/2015
975                              <1>       ; 18/10/2014
976                              <1>       ; 16/10/2014 (Retro UNIX 386 v1 - beginning)
977                              <1>       ;
978                              <1>       ; INPUT ->
979                              <1>       ;     [u.pgdir] = PHYSICAL (real/flat) ADDRESS of the parent's
980                              <1>       ;                 page directory.
981                              <1>       ; OUTPUT ->
982                              <1>       ;     EAX =  PHYSICAL (real/flat) ADDRESS of the child's
983                              <1>       ;            page directory.
984                              <1>       ;     (New page directory with new page table entries.)
985                              <1>       ;     (New page tables with read only copies of the parent's
986                              <1>       ;     pages.)
987                              <1>       ;     EAX = 0 -> Error (CF = 1)
988                              <1>       ;
989                              <1>       ; Modified Registers -> none (except EAX)
990                              <1>       ;
991 00005DD8 E8ECFDFFFF          <1>       call   allocate_page
```

```
 992 00005DDD 723E                <1>       jc     short dpd_err
 993                              <1>       ;
 994 00005DDF 55                  <1>       push   ebp ; 20/07/2015
 995 00005DE0 56                  <1>       push   esi
 996 00005DE1 57                  <1>       push   edi
 997 00005DE2 53                  <1>       push   ebx
 998 00005DE3 51                  <1>       push   ecx
 999 00005DE4 8B35[B8030300]      <1>       mov    esi, [u.pgdir]
1000 00005DEA 89C7                <1>       mov    edi, eax
1001 00005DEC 50                  <1>       push   eax ; save child's page directory address
1002                              <1>       ; 31/08/2015
1003                              <1>       ; copy PDE 0 from the parent's page dir to the child's page dir
1004                              <1>       ; (use same system space for all user page tables)
1005 00005DED A5                  <1>       movsd
1006 00005DEE BD00004000          <1>       mov    ebp, 1024*4096 ; pass the 1st 4MB (system space)
1007 00005DF3 B9FF030000          <1>       mov    ecx, (PAGE_SIZE / 4) - 1 ; 1023
1008                              <1> dpd_0:
1009 00005DF8 AD                  <1>       lodsd
1010                              <1>       ;or    eax, eax
1011                              <1>         ;jnz     short dpd_1
1012 00005DF9 A801                <1>       test   al, PDE_A_PRESENT ;  bit 0 =  1
1013 00005DFB 7508                <1>       jnz    short dpd_1
1014                              <1>       ; 20/07/2015 (virtual address at the end of the page table)
1015 00005DFD 81C500004000        <1>       add    ebp, 1024*4096 ; page size * PTE count
1016 00005E03 EB0F                <1>       jmp    short dpd_2
1017                              <1> dpd_1:
1018 00005E05 662500F0            <1>       and    ax, PDE_A_CLEAR ; 0F000h ; clear attribute bits
1019 00005E09 89C3                <1>       mov    ebx, eax
1020                              <1>       ; EBX = Parent's page table address
1021 00005E0B E81F000000          <1>       call   duplicate_page_table
1022 00005E10 720C                <1>       jc     short dpd_p_err
1023                              <1>       ; EAX = Child's page table address
1024 00005E12 0C07                <1>       or     al, PDE_A_PRESENT + PDE_A_WRITE + PDE_A_USER
1025                              <1>                    ; set bit 0, bit 1 and bit 2 to 1
1026                              <1>                    ; (present, writable, user)
1027                              <1> dpd_2:
1028 00005E14 AB                  <1>       stosd
1029 00005E15 E2E1                <1>       loop   dpd_0
1030                              <1>       ;
1031 00005E17 58                  <1>       pop    eax  ; restore child's page directory address
1032                              <1> dpd_3:
1033 00005E18 59                  <1>       pop    ecx
1034 00005E19 5B                  <1>       pop    ebx
1035 00005E1A 5F                  <1>       pop    edi
1036 00005E1B 5E                  <1>       pop    esi
1037 00005E1C 5D                  <1>       pop    ebp ; 20/07/2015
1038                              <1> dpd_err:
1039 00005E1D C3                  <1>       retn
1040                              <1> dpd_p_err:
1041                              <1>       ; release the allocated pages missing (recover free space)
1042 00005E1E 58                  <1>       pop    eax  ; the new page directory address (physical)
1043 00005E1F 8B1D[B8030300]      <1>       mov    ebx, [u.pgdir] ; parent's page directory address
1044 00005E25 E8D8FEFFFF          <1>       call   deallocate_page_dir
1045 00005E2A 29C0                <1>       sub    eax, eax ; 0
1046 00005E2C F9                  <1>       stc
1047 00005E2D EBE9                <1>       jmp    short dpd_3
1048                              <1>
1049                              <1> duplicate_page_table:
1050                              <1>       ; 20/02/2017
1051                              <1>       ; 21/09/2015
1052                              <1>       ; 20/07/2015
1053                              <1>       ; 05/05/2015
1054                              <1>       ; 28/04/2015
1055                              <1>       ; 27/04/2015
1056                              <1>       ; 18/04/2015
1057                              <1>       ; 18/10/2014
1058                              <1>       ; 16/10/2014 (Retro UNIX 386 v1 - beginning)
1059                              <1>       ;
1060                              <1>       ; INPUT ->
1061                              <1>       ;      EBX = PHYSICAL (real/flat) ADDRESS of the parent's page table.
1062                              <1>       ;        20/02/2017
1063                              <1>       ;      EBP = Linear address of the page (from 'duplicate_page_dir')
1064                              <1>       ;            (Linear address = CORE + user's virtual address)
1065                              <1>       ; OUTPUT ->
1066                              <1>       ;      EAX = PHYSICAL (real/flat) ADDRESS of the child's page table.
1067                              <1>       ;            (with 'read only' attribute of page table entries)
1068                              <1>       ;      20/02/2017
1069                              <1>       ;      EBP = Next linear page address (for 'duplicate_page_dir')
1070                              <1>       ;
1071                              <1>       ;      CF = 1 -> error
1072                              <1>       ;
1073                              <1>       ; Modified Registers -> EBP (except EAX)
1074                              <1>       ;
1075 00005E2F E895FDFFFF          <1>       call   allocate_page
1076 00005E34 726A                <1>       jc     short dpt_err
1077                              <1>       ;
1078 00005E36 50                  <1>       push   eax ; *
1079 00005E37 56                  <1>       push   esi
1080 00005E38 57                  <1>       push   edi
1081 00005E39 52                  <1>       push   edx
1082 00005E3A 51                  <1>       push   ecx
1083                              <1>       ;
1084 00005E3B 89DE                <1>       mov    esi, ebx
1085 00005E3D 89C7                <1>       mov    edi, eax
1086 00005E3F 89C2                <1>       mov    edx, eax
1087 00005E41 81C200100000        <1>       add    edx, PAGE_SIZE
1088                              <1> dpt_0:
1089 00005E47 AD                  <1>       lodsd
1090 00005E48 21C0                <1>       and    eax, eax
1091 00005E4A 7444                <1>       jz     short dpt_3
1092 00005E4C A801                <1>       test   al, PTE_A_PRESENT ;  bit 0 =  1
1093 00005E4E 7507                <1>       jnz    short dpt_1
1094                              <1>       ; 20/07/2015
1095                              <1>       ; ebp = virtual (linear) address of the memory page
1096 00005E50 E83F050000          <1>       call   reload_page ; 28/04/2015
```

```
1097 00005E55 7244              <1>       jc      short dpt_p_err
1098                            <1> dpt_1:
1099                            <1>       ; 21/09/2015
1100 00005E57 89C1              <1>       mov     ecx, eax
1101 00005E59 662500F0          <1>       and     ax, PTE_A_CLEAR ; 0F000h ; clear attribute bits
1102 00005E5D F6C102            <1>       test    cl, PTE_A_WRITE ; writable page ?
1103 00005E60 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 00005E62 E862FDFFFF        <1>       call    allocate_page
1108 00005E67 7232              <1>       jc      short dpt_p_err
1109 00005E69 57                <1>       push    edi
1110 00005E6A 56                <1>       push    esi
1111 00005E6B 89CE              <1>       mov     esi, ecx
1112 00005E6D 89C7              <1>       mov     edi, eax
1113 00005E6F B900040000        <1>       mov     ecx, PAGE_SIZE/4
1114 00005E74 F3A5              <1>       rep     movsd  ; copy page (4096 bytes)
1115 00005E76 5E                <1>       pop     esi
1116 00005E77 5F                <1>       pop     edi
1117                            <1>       ;
1118 00005E78 53                <1>       push    ebx
1119 00005E79 50                <1>       push    eax
1120                            <1>       ; 20/07/2015
1121 00005E7A 89EB              <1>       mov     ebx, ebp
1122                            <1>       ; ebx = virtual (linear) address of the memory page
1123 00005E7C E887030000        <1>       call    add_to_swap_queue
1124 00005E81 58                <1>       pop     eax
1125 00005E82 5B                <1>       pop     ebx
1126                            <1>       ; 21/09/2015
1127 00005E83 0C07              <1>       or      al, PTE_A_USER+PTE_A_WRITE+PTE_A_PRESENT
1128                            <1>               ; user + writable + present page
1129 00005E85 EB09              <1>       jmp     short dpt_3
1130                            <1> dpt_2:
1131                            <1>       ;or     ax, PTE_A_USER+PTE_A_PRESENT
1132 00005E87 0C05              <1>       or      al, PTE_A_USER+PTE_A_PRESENT
1133                            <1>                   ; (read only page!)
1134 00005E89 8946FC            <1>       mov     [esi-4], eax ; update parent's PTE
1135 00005E8C 660D0002          <1>       or       ax, PTE_DUPLICATED  ; (read only page & duplicated PTE!)
1136                            <1> dpt_3:
1137 00005E90 AB                <1>       stosd  ; EDI points to child's PTE
1138                            <1>       ;
1139 00005E91 81C500100000      <1>       add     ebp, 4096 ; 20/07/2015 (next page)
1140                            <1>       ;
1141 00005E97 39D7              <1>       cmp     edi, edx
1142 00005E99 72AC              <1>       jb      short dpt_0
1143                            <1> dpt_p_err:
1144 00005E9B 59                <1>       pop     ecx
1145 00005E9C 5A                <1>       pop     edx
1146 00005E9D 5F                <1>       pop     edi
1147 00005E9E 5E                <1>       pop     esi
1148 00005E9F 58                <1>       pop     eax ; *
1149                            <1> dpt_err:
1150 00005EA0 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 00005EA1 53                       <1>       push   ebx
1293 00005EA2 52                       <1>       push   edx
1294 00005EA3 51                       <1>       push   ecx
1295                                    <1>       ;
1296                                    <1>       ; 21/09/2015 (debugging)
1297 00005EA4 FF05[CC030300]           <1>       inc    dword [u.pfcount] ; page fault count for running process
1298 00005EAA 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 00005EB0 8A15[78050300]           <1>       mov    dl, [error_code]
1302                                    <1>       ;
1303 00005EB6 F6C201                   <1>       test   dl, 1  ; page fault was caused by a non-present page
1304                                    <1>                     ; sign
1305 00005EB9 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 00005EBB F6C202             <1>       test   dl, 2  ; page fault was caused by a page write
1328                            <1>                     ; sign
1329 00005EBE 0F84AB000000     <1>         jz      pfh_p_err
1330                            <1>       ; 31/08/2015
1331 00005EC4 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 00005EC7 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 00005ECD 0F20D3           <1>       mov    ebx, cr2 ; CR2 contains the linear address
1339                            <1>                     ; which has caused to page fault
1340 00005ED0 E8A2000000       <1>       call   copy_page
1341 00005ED5 0F828D000000     <1>         jc      pfh_im_err ; insufficient memory
1342                            <1>       ;
1343 00005EDB EB7D             <1>         jmp     pfh_cpp_ok
1344                            <1>       ;
1345                            <1> pfh_alloc_np:
1346 00005EDD E8E7FCFFFF       <1>       call   allocate_page; (allocate a new page)
1347 00005EE2 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 00005EE8 80E204             <1>       and    dl, 4  ; CPL = 3 ?
1353 00005EEB 7505             <1>       jnz    short pfh_um
1354                            <1>                     ; Page fault handler for kernel/system mode (CPL=0)
1355 00005EED 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 00005EF0 EB06             <1>       jmp    short pfh_get_pde
1360                            <1>       ;
1361                            <1> pfh_um:                   ; Page fault handler for user/appl. mode (CPL=3)
1362 00005EF2 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 00005EF8 80CA03           <1>       or     dl, 3  ; USER + WRITE + PRESENT or SYSTEM + WRITE + PRESENT
1367 00005EFB 0F20D1           <1>       mov    ecx, cr2 ; CR2 contains the virtual address
1368                            <1>                      ; which has been caused to page fault
1369                            <1>                      ;
1370 00005EFE C1E914           <1>       shr    ecx, 20       ; shift 20 bits right
1371 00005F01 80E1FC           <1>       and    cl, 0FCh ; mask lower 2 bits to get PDE offset
1372                            <1>       ;
1373 00005F04 01CB             <1>       add    ebx, ecx ; now, EBX points to the relevant page dir entry
1374 00005F06 8B0B             <1>       mov    ecx, [ebx] ; physical (base) address of the page table
1375 00005F08 F6C101           <1>       test   cl, 1   ; check bit 0 is set (1) or not (0).
1376 00005F0B 740B             <1>       jz     short pfh_set_pde ; Page directory entry is not valid,
1377                            <1>                              ; set/validate page directory entry
1378 00005F0D 6681E100F0       <1>       and    cx, PDE_A_CLEAR ; 0F000h ; Clear attribute bits
1379 00005F12 89CB             <1>       mov    ebx, ecx ; Physical address of the page table
1380 00005F14 89C1             <1>       mov    ecx, eax ; new page address (physical)
1381 00005F16 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 00005F18 08D0             <1>       or     al, dl  ; lower 3 bits are used as U/S, R/W, P flags
1387 00005F1A 8903             <1>       mov    [ebx], eax ; Let's put the new page directory entry here !
1388 00005F1C 30C0             <1>       xor    al, al  ; clear lower (3..8) bits
1389 00005F1E 89C3             <1>       mov    ebx, eax
1390 00005F20 E8A4FCFFFF       <1>       call   allocate_page ; (allocate a new page)
1391 00005F25 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 00005F27 89C1             <1>       mov    ecx, eax
1395 00005F29 E815FDFFFF       <1>       call   clear_page ; Clear page content
1396                            <1> pfh_get_pte:
1397 00005F2E 0F20D0           <1>       mov    eax, cr2 ; virtual address
1398                            <1>                       ; which has been caused to page fault
1399 00005F31 89C7             <1>       mov    edi, eax ; 20/07/2015
1400 00005F33 C1E80C           <1>       shr    eax, 12       ; shift 12 bit right to get
1401                            <1>                      ; higher 20 bits of the page fault address
1402 00005F36 25FF030000       <1>       and    eax, 3FFh ; mask PDE# bits, the result is PTE# (0 to 1023)
1403 00005F3B C1E002           <1>       shl    eax, 2 ; shift 2 bits left to get PTE offset
1404 00005F3E 01C3             <1>       add    ebx, eax ; now, EBX points to the relevant page table entry
1405 00005F40 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 00005F42 21C0             <1>       and    eax, eax
1408 00005F44 7410             <1>       jz     short pfh_gpte_1
1409                            <1>       ; 20/07/2015
1410 00005F46 87D9             <1>       xchg   ebx, ecx ; new page address (physical)
1411 00005F48 55               <1>       push   ebp ; 20/07/2015
```

```
1412 00005F49 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 00005F4C E8B7000000          <1>       call  swap_in
1418 00005F51 5D                  <1>       pop   ebp
1419 00005F52 7210                <1>       jc      short pfh_err_retn
1420 00005F54 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 00005F56 08D1                <1>       or    cl, dl ; lower 3 bits are used as U/S, R/W, P flags
1425 00005F58 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 00005F5A 0F20D3              <1>       mov   ebx, cr2
1429 00005F5D 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 00005F62 31C0                <1>       xor   eax, eax  ; 0
1438                              <1>       ;
1439                              <1> pfh_err_retn:
1440 00005F64 59                  <1>       pop   ecx
1441 00005F65 5A                  <1>       pop   edx
1442 00005F66 5B                  <1>       pop   ebx
1443 00005F67 C3                  <1>       retn
1444                              <1>
1445                              <1> pfh_im_err:
1446 00005F68 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 00005F6D 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 00005F6F 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 00005F74 F9                  <1>       stc
1459 00005F75 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 00005F77 56                  <1>       push  esi
1489 00005F78 57                  <1>       push  edi
1490                              <1>       ;push ebx
1491                              <1>       ;push ecx
1492 00005F79 31F6                <1>       xor   esi, esi
1493 00005F7B C1EB0C              <1>       shr   ebx, 12 ; shift 12 bits right to get PDE & PTE numbers
1494 00005F7E 89D9                <1>       mov   ecx, ebx ; save page fault address (as 12 bit shifted)
1495 00005F80 C1EB08              <1>       shr   ebx, 8  ; shift 8 bits right and then
1496 00005F83 80E3FC              <1>       and   bl, 0FCh ; mask lower 2 bits to get PDE offset
1497 00005F86 89DF                <1>       mov   edi, ebx ; save it for the parent of current process
1498 00005F88 031D[B8030300]      <1>       add   ebx, [u.pgdir] ; EBX points to the relevant page dir entry
1499 00005F8E 8B03                <1>       mov   eax, [ebx] ; physical (base) address of the page table
1500 00005F90 662500F0            <1>       and   ax, PTE_A_CLEAR ; 0F000h ; clear attribute bits
1501 00005F94 89CB                <1>       mov   ebx, ecx   ; (restore higher 20 bits of page fault address)
1502 00005F96 81E3FF030000        <1>       and   ebx, 3FFh  ; mask PDE# bits, the result is PTE# (0 to 1023)
1503 00005F9C 66C1E302            <1>       shl   bx, 2    ; shift 2 bits left to get PTE offset
1504 00005FA0 01C3                <1>       add   ebx, eax   ; EBX points to the relevant page table entry
1505                              <1>       ; 07/09/2015
1506 00005FA2 66F7030002          <1>         test    word [ebx], PTE_DUPLICATED ; (Does current process share this
1507                              <1>                                 ; read only page as a child process?)
1508 00005FA7 7509                <1>       jnz   short cpp_0 ; yes
1509 00005FA9 8B0B                <1>       mov   ecx, [ebx] ; PTE value
1510 00005FAB 6681E100F0          <1>       and   cx, PTE_A_CLEAR ; 0F000h  ; clear page attributes
1511 00005FB0 EB32                <1>       jmp   short cpp_1
1512                              <1> cpp_0:
1513 00005FB2 89FE                <1>       mov   esi, edi
1514 00005FB4 0335[BC030300]      <1>       add   esi, [u.ppgdir] ; the parent's page directory entry
1515 00005FBA 8B06                <1>       mov   eax, [esi] ; physical (base) address of the page table
1516 00005FBC 662500F0            <1>       and   ax, PTE_A_CLEAR ; 0F000h ; clear attribute bits
```

```
1517 00005FC0 89CE              <1>       mov   esi, ecx   ; (restore higher 20 bits of page fault address)
1518 00005FC2 81E6FF030000      <1>       and   esi, 3FFh  ; mask PDE# bits, the result is PTE# (0 to 1023)
1519 00005FC8 66C1E602          <1>       shl   si, 2    ; shift 2 bits left to get PTE offset
1520 00005FCC 01C6              <1>       add   esi, eax   ; EDX points to the relevant page table entry
1521 00005FCE 8B0E              <1>       mov   ecx, [esi] ; PTE value of the parent process
1522                            <1>       ; 21/09/2015
1523 00005FD0 8B03              <1>       mov   eax, [ebx] ; PTE value of the child process
1524 00005FD2 662500F0          <1>       and   ax, PTE_A_CLEAR ; 0F000h ; clear page attributes
1525                            <1>       ;
1526 00005FD6 F6C101            <1>       test  cl, PTE_A_PRESENT ; is it a present/valid page ?
1527 00005FD9 7424              <1>       jz    short cpp_3 ; the parent's page is not same page
1528                            <1>       ;
1529 00005FDB 6681E100F0        <1>       and   cx, PTE_A_CLEAR ; 0F000h ; clear page attributes
1530 00005FE0 39C8              <1>       cmp   eax, ecx   ; Same page?
1531 00005FE2 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 00005FE4 E8E0FBFFFF        <1>       call  allocate_page
1535 00005FE9 721A              <1>       jc    short cpp_4 ; 'insufficient memory' error
1536 00005FEB 21F6              <1>       and   esi, esi    ; check ESI is valid or not
1537 00005FED 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 00005FEF 890E              <1>       mov   [esi], ecx
1543 00005FF1 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 00005FF4 89C7              <1>       mov   edi, eax ; new page address of the child process
1547                            <1>       ; 07/09/2015
1548 00005FF6 89CE              <1>       mov   esi, ecx ; the page address of the parent process
1549 00005FF8 B900040000        <1>       mov   ecx, PAGE_SIZE / 4
1550 00005FFD F3A5              <1>       rep   movsd ; 31/08/2015
1551                            <1> cpp_3:
1552 00005FFF 0C07              <1>       or    al, PTE_A_PRESENT + PTE_A_WRITE + PTE_A_USER ; 1+2+4 = 7
1553 00006001 8903              <1>       mov   [ebx], eax ; Update PTE
1554 00006003 28C0              <1>       sub   al, al ; clear attributes
1555                            <1> cpp_4:
1556                            <1>       ;pop  ecx
1557                            <1>       ;pop  ebx
1558 00006005 5F                <1>       pop   edi
1559 00006006 5E                <1>       pop   esi
1560 00006007 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
1605                            <1> ;;    the check, if [swpq_count] < 1023. If [swpq_count] = 1024
1606                            <1> ;;    the queue will be rotated and the PTE in the head will be
1607                            <1> ;;    added to the tail after resetting 'accessed' bit.
1608                            <1> ;;
1609                            <1> ;;
1610                            <1> ;;
1611                            <1> ;; SWAP DISK/FILE (with 4096 bytes swapped page blocks)
1612                            <1> ;;
1613                            <1> ;;  00000000  00000004  00000008  0000000C   ...   size-8    size-4
1614                            <1> ;; +---------+---------+---------+---------+-- --+---------+---------+
1615                            <1> ;; |descriptr| page(1) | page(2) | page(3) | ... |page(n-1)| page(n) |
1616                            <1> ;; +---------+---------+---------+---------+-- --+---------+---------+
1617                            <1> ;;
1618                            <1> ;; [swpd_next] = the first free block address in swapped page records
1619                            <1> ;;                  for next free block search by 'swap_out' procedure.
1620                            <1> ;; [swpd_size] = swap disk/file size in sectors (512 bytes)
1621                            <1> ;;           NOTE: max. possible swap disk size is 1024 GB
```

```
1622                    <1> ;;           (entire swap space must be accessed by using
1623                    <1> ;;           31 bit offset address)
1624                    <1> ;; [swpd_free] = free block (4096 bytes) count in swap disk/file space
1625                    <1> ;; [swpd_start] = absolute/start address of the swap disk/file
1626                    <1> ;;            0 for file, or beginning sector of the swap partition
1627                    <1> ;; [swp_drv] = logical drive description table addr. of swap disk/file
1628                    <1> ;;
1629                    <1> ;;
1630                    <1> ;; Method:
1631                    <1> ;;    When the memory (ram) becomes insufficient, page allocation
1632                    <1> ;;    procedure swaps out a page from memory to the swap disk
1633                    <1> ;;    (partition) or swap file to get a new free page at the memory.
1634                    <1> ;;    Swapping out is performed by using swap page queue.
1635                    <1> ;;
1636                    <1> ;;    Allocation block size of swap disk/file is equal to page size
1637                    <1> ;;    (4096 bytes). Swapping address (in sectors) is recorded
1638                    <1> ;;    into relevant page file entry as 31 bit physical (logical)
1639                    <1> ;;    offset address as 1 bit shifted to left for present flag (0).
1640                    <1> ;;    Swapped page address is between 1 and swap disk/file size - 4.
1641                    <1> ;;    Absolute physical (logical) address of the swapped page is
1642                    <1> ;;    calculated by adding offset value to the swap partition's
1643                    <1> ;;    start address. If the swap device (disk) is a virtual disk
1644                    <1> ;;    or it is a file, start address of the swap disk/volume is 0,
1645                    <1> ;;    and offset value is equal to absolute (physical or logical)
1646                    <1> ;;    address/position. (It has not to be ZERO if the swap partition
1647                    <1> ;;    is in a partitioned virtual hard disk.)
1648                    <1> ;;
1649                    <1> ;;    Note: Swap addresses are always specified/declared in sectors,
1650                    <1> ;;    not in bytes or     in blocks/zones/clusters (4096 bytes) as unit.
1651                    <1> ;;
1652                    <1> ;;    Swap disk/file allocation is mapped via 'Swap Allocation Table'
1653                    <1> ;;    at memory as similar to 'Memory Allocation Table'.
1654                    <1> ;;
1655                    <1> ;;    Every bit of Swap Allocation Table repsesents one swap block
1656                    <1> ;;    (equal to page size) respectively. Bit 0 of the S.A.T. byte 0
1657                    <1> ;;    is reserved for swap disk/file block 0 as descriptor block
1658                    <1> ;;    (also for compatibility with PTE). If bit value is ZERO,
1659                    <1> ;;    it means relevant (respective) block is in use, and,
1660                    <1> ;;    of course, if bit value is 1, it means relevant (respective)
1661                    <1> ;;      swap disk/file block is free.
1662                    <1> ;;    For example: bit 1 of the byte 128 repsesents block 1025
1663                    <1> ;;    (128*8+1) or sector (offset) 8200 on the swap disk or
1664                    <1> ;;    byte (offset/position) 4198400 in the swap file.
1665                    <1> ;;    4GB swap space is represented via 128KB Swap Allocation Table.
1666                    <1> ;;    Initial layout of Swap Allocation Table is as follows:
1667                    <1> ;;    ------------------------------------------------------------
1668                    <1> ;;    01111111111111111111111 .... 11111111111111111111111111111
1669                    <1> ;;    ------------------------------------------------------------
1670                    <1> ;;    (0 is reserved block, 1s represent free blocks respectively.)
1671                    <1> ;;    (Note: Allocation cell/unit of the table is bit, not byte)
1672                    <1> ;;
1673                    <1> ;;    ............................................................
1674                    <1> ;;
1675                    <1> ;;    'swap_out' procedure checks 'free_swap_blocks' count at first,
1676                    <1> ;;    then it searches Swap Allocation Table if free count is not
1677                    <1> ;;    zero. From begining the [swpd_next] dword value, the first bit
1678                    <1> ;;    position with value of 1 on the table is converted to swap
1679                    <1> ;;    disk/file offset address, in sectors (not 4096 bytes block).
1680                    <1> ;;    'ldrv_write' procedure is called with ldrv (logical drive
1681                    <1> ;;    number of physical swap disk or virtual swap disk)
1682                    <1> ;;    number, sector offset (not absolute sector -LBA- number),
1683                    <1> ;;    and sector count (8, 512*8 = 4096) and buffer adress
1684                    <1> ;;    (memory page). That will be a direct disk write procedure.
1685                    <1> ;;    (for preventing late memory allocation, significant waiting).
1686                    <1> ;;    If disk write procedure returns with error or free count of
1687                    <1> ;;    swap blocks is ZERO, 'swap_out' procedure will return with
1688                    <1> ;;    'insufficient memory error' (cf=1).
1689                    <1> ;;
1690                    <1> ;;    (Note: Even if free swap disk/file blocks was not zero,
1691                    <1> ;;    any disk write error will not be fixed by 'swap_out' procedure,
1692                    <1> ;;    in other words, 'swap_out' will not check the table for other
1693                    <1> ;;    free blocks after a disk write error. It will return to
1694                    <1> ;;    the caller with error (CF=1) which means swapping is failed.
1695                    <1> ;;
1696                    <1> ;;    After writing the page on to swap disk/file address/sector,
1697                    <1> ;;    'swap_out' procesure returns with that swap (offset) sector
1698                    <1> ;;    address (cf=0).
1699                    <1> ;;
1700                    <1> ;;    ............................................................
1701                    <1> ;;
1702                    <1> ;;    'swap_in' procedure loads addressed (relevant) swap disk or
1703                    <1> ;;    file sectors at specified memory page. Then page allocation
1704                    <1> ;;    procedure updates relevant page table entry with 'present'
1705                    <1> ;;    attribute. If swap disk or file reading fails there is nothing
1706                    <1> ;;    to do, except to terminate the process which is the owner of
1707                    <1> ;;    the swapped page.
1708                    <1> ;;
1709                    <1> ;;    'swap_in' procedure sets the relevant/respective bit value
1710                    <1> ;;    in the Swap Allocation Table (as free block). 'swap_in' also
1711                    <1> ;;    updates [swpd_first] pointer if it is required.
1712                    <1> ;;
1713                    <1> ;;    ............................................................
1714                    <1> ;;
1715                    <1> ;;    Note: If [swap_enabled] value is ZERO, that means there is not
1716                    <1> ;;    a swap disk or swap file in use... 'swap_in' and 'swap_out'
1717                    <1> ;;    procedures ans 'swap page que' procedures will not be active...
1718                    <1> ;;    'Insufficient memory' error will be returned by 'swap_out'
1719                    <1> ;;    and 'general protection fault' will be returned by 'swap_in'
1720                    <1> ;;    procedure, if it is called mistakenly (a wrong value in a PTE).
1721                    <1> ;;
1722                    <1>
1723                    <1> swap_in:
1724                    <1>       ; 31/08/2015
1725                    <1>       ; 20/07/2015
1726                    <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>
00006008 833D[62050300]00       <1>         cmp     dword [swp_drv], 0
0000600F 7646                   <1>       jna     short swpin_dnp_err
                                <1>
00006011 3B05[66050300]         <1>       cmp     eax, [swpd_size]
00006017 734A                   <1>       jnb     short swpin_snp_err
                                <1>
00006019 56                     <1>       push    esi
0000601A 53                     <1>       push    ebx
0000601B 51                     <1>       push    ecx
0000601C 8B35[62050300]         <1>       mov     esi, [swp_drv]
00006022 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
00006027 50                     <1>       push    eax
00006028 E8AF020000             <1>       call    logical_disk_read
0000602D 58                     <1>       pop     eax
0000602E 730C                   <1>       jnc     short swpin_read_ok
                                <1>       ;
00006030 B828000000             <1>       mov     eax, SWP_DISK_READ_ERR ; drive not ready or read error
00006035 A3[C8030300]           <1>       mov     [u.error], eax
0000603A EB17                   <1>       jmp     short swpin_retn
                                <1>       ;
                                <1> swpin_read_ok:
                                <1>       ; EAX = Offset address (logical sector number)
0000603C E80D020000             <1>       call    unlink_swap_block  ; Deallocate swap block
                                <1>       ;
                                <1>       ; EBX = Memory page (buffer) address (physical!)
                                <1>       ; 20/07/2015
00006041 89EB                   <1>       mov     ebx, ebp ; virtual address (page fault address)
00006043 6681E300F0             <1>         and     bx, ~PAGE_OFF ; ~0FFFh ; reset bits, 0 to 11
00006048 8A1D[B3030300]         <1>       mov     bl, [u.uno] ; current process number
                                <1>       ; EBX = Virtual (Linear) address & process number combination
0000604E 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:
00006053 59                     <1>       pop     ecx
00006054 5B                     <1>       pop     ebx
00006055 5E                     <1>       pop     esi
00006056 C3                     <1>       retn
                                <1>
                                <1> swpin_dnp_err:
00006057 B829000000             <1>       mov     eax, SWP_DISK_NOT_PRESENT_ERR
                                <1> swpin_err_retn:
0000605C A3[C8030300]           <1>       mov     [u.error], eax
00006061 F9                     <1>       stc
00006062 C3                     <1>       retn
                                <1>
                                <1> swpin_snp_err:
00006063 B82A000000             <1>       mov     eax, SWP_SECTOR_NOT_PRESENT_ERR
00006068 EBF2                   <1>       jmp     short swpin_err_retn
                                <1>
                                <1> swap_out:
                                <1>       ; 10/06/2016
                                <1>       ; 07/06/2016
                                <1>         ; 23/05/2016
                                <1>       ; 19/05/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>       ; 24/10/2014 - 31/08/2015 (Retro UNIX 386 v1)
                                <1>       ;
                                <1>       ; INPUT ->
                                <1>       ;     none
                                <1>       ;
                                <1>       ; OUTPUT ->
                                <1>       ;     EAX = Physical page address (which is swapped out
                                <1>       ;           for allocating a new page)
                                <1>       ;     CF = 1 -> swap disk writing error (disk/file not present
                                <1>       ;              or sector not present or drive not ready
                                <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 -> none (except EAX)
                                <1>       ;
0000606A 66833D[60050300]01     <1>       cmp     word [swpq_count], 1
```

```
1832 00006072 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 00006078 833D[6A050300]01    <1>         cmp     dword [swpd_free], 1
1838 0000607F 0F828F000000        <1>         jc      swpout_nfspc_err ; 'no free space on swap disk'
1839                              <1>
1840 00006085 53                  <1>       push   ebx ; *
1841                              <1> swpout_1:
1842                              <1>       ; 10/06/2016
1843 00006086 31DB                <1>       xor    ebx, ebx ; shift the queue and return a PTE value
1844 00006088 E8A1000000          <1>       call   swap_queue_shift
1845 0000608D 21C0                <1>       and    eax, eax     ; 0 = empty queue (improper entries)
1846 0000608F 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 00006095 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 00006099 52                  <1>       push   edx ; **
1858                              <1>
1859                              <1>       ; 07/06/2016
1860 0000609A 803D[238E0100]00    <1>       cmp    byte [timer_events], 0
1861 000060A1 762F                <1>       jna    short swpout_2
1862                              <1>       ;
1863 000060A3 8A15[238E0100]      <1>       mov    dl, [timer_events]
1864                              <1>
1865 000060A9 51                  <1>       push   ecx ; ***
1866 000060AA 53                  <1>       push   ebx ; ****
1867 000060AB BB[60040300]        <1>       mov    ebx, timer_set ; beginning address of timer event
1868                              <1>                               ; structures
1869                              <1> swpout_timer_page_1:
1870 000060B0 8A0B                <1>       mov    cl, [ebx]
1871 000060B2 08C9                <1>       or     cl, cl ; 0 = free, >0 = process number
1872 000060B4 7415                <1>       jz     short swpout_timer_page_3
1873 000060B6 8B4B0C              <1>       mov    ecx, [ebx+12] ; response (signal return) address
1874 000060B9 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 000060BE 39C8                <1>       cmp    eax, ecx
1878 000060C0 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 000060C2 5B                  <1>       pop    ebx ; ****
1890 000060C3 59                  <1>       pop    ecx ; ***
1891 000060C4 5A                  <1>       pop    edx ; **
1892 000060C5 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 000060C7 FECA                <1>       dec    dl
1897 000060C9 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 000060CB 83C310              <1>       add    ebx, 16
1902 000060CE EBE0                <1>       jmp    short swpout_timer_page_1
1903                              <1>
1904                              <1> swpout_timer_page_4:
1905 000060D0 5B                  <1>       pop    ebx ; ****
1906 000060D1 59                  <1>       pop    ecx ; ***
1907                              <1> swpout_2:
1908 000060D2 89DA                <1>       mov    edx, ebx            ; Page table entry address
1909 000060D4 89C3                <1>       mov    ebx, eax            ; Buffer (Page) Address
1910                              <1>       ;
1911 000060D6 E8A6010000          <1>       call   link_swap_block
1912 000060DB 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 000060DD 5A                  <1>       pop    edx ; **
1916 000060DE 5B                  <1>       pop    ebx ; *
1917 000060DF EB33                <1>       jmp    short swpout_nfspc_err
1918                              <1> swpout_3:
1919 000060E1 A900000080          <1>       test   eax, 80000000h ; test bit 31 (this may not be needed!)
1920 000060E6 752C                <1>       jnz    short swpout_nfspc_err  ; 10/06/2016 (bit 31 = 1 !)
1921                              <1>       ;
1922 000060E8 56                  <1>       push   esi ; **
1923 000060E9 51                  <1>       push   ecx ; ***
1924 000060EA 50                  <1>       push   eax ; sector address ; (31 bit !, bit 31 = 0)
1925 000060EB 8B35[62050300]      <1>       mov    esi, [swp_drv]
1926 000060F1 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 000060F6 E8E2010000          <1>       call   logical_disk_write
1938                              <1>       ; edx = PTE address
1939 000060FB 59                  <1>       pop    ecx ; sector address
1940 000060FC 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 000060FE B82C000000          <1>       mov    eax, SWP_DISK_WRITE_ERR ; drive not ready or write error
1945 00006103 A3[C8030300]        <1>       mov    [u.error], eax
1946 00006108 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 0000610A D1E1                <1>       shl    ecx, 1  ; 31 bit sector address from bit 1 to bit 31
1953 0000610C 890A                <1>       mov    [edx], ecx
1954                              <1>              ; bit 0 = 0 (swapped page)
1955 0000610E 89D8                <1>       mov    eax, ebx
1956                              <1> swpout_retn:
1957 00006110 59                  <1>       pop    ecx ; ***
1958 00006111 5E                  <1>       pop    esi ; **
1959 00006112 5B                  <1>       pop    ebx ; *
1960 00006113 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 00006114 B82B000000          <1>       mov    eax, SWP_NO_FREE_SPACE_ERR ; no free space
1967                              <1> swpout_err_retn:
1968 00006119 A3[C8030300]        <1>       mov    [u.error], eax
1969                              <1>       ;stc
1970 0000611E C3                  <1>       retn
1971                              <1> swpout_npts_err:
1972 0000611F B82D000000          <1>       mov    eax, SWP_NO_PAGE_TO_SWAP_ERR
1973 00006124 5B                  <1>       pop    ebx
1974 00006125 EBF2                <1>       jmp    short swpout_err_retn
1975                              <1> swpout_im_err:
1976 00006127 B804000000          <1>       mov    eax, ERR_MINOR_IM ; insufficient (out of) memory
1977 0000612C 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 0000612E 0FB705[60050300]    <1>       movzx   eax, word [swpq_count]  ; Max. 1024
2031 00006135 6621C0              <1>       and    ax, ax
2032 00006138 7431                <1>       jz     short swpqs_retn
2033 0000613A 57                  <1>       push   edi
2034 0000613B 56                  <1>       push   esi
2035 0000613C 51                  <1>       push   ecx
2036 0000613D BE00E00800          <1>       mov    esi, swap_queue
2037 00006142 89C1                <1>       mov    ecx, eax
2038 00006144 09DB                <1>       or     ebx, ebx
2039 00006146 7424                <1>       jz     short swpqs_7
2040                              <1> swpqs_1:
2041 00006148 AD                  <1>       lodsd
```

```
2042 00006149 39D8                <1>       cmp    eax, ebx
2043 0000614B 7406                <1>       je     short swpqs_2
2044 0000614D E2F9                <1>       loop   swpqs_1
2045                              <1>       ; 10/06/2016
2046 0000614F 29C0                <1>       sub    eax, eax
2047 00006151 EB15                <1>       jmp    short swpqs_6
2048                              <1> swpqs_2:
2049 00006153 89F7                <1>       mov    edi, esi
2050 00006155 83EF04              <1>       sub    edi, 4
2051                              <1> swpqs_3:
2052 00006158 66FF0D[60050300]    <1>       dec    word [swpq_count]
2053 0000615F 7403                <1>       jz     short swpqs_5
2054                              <1> swpqs_4:
2055 00006161 49                  <1>       dec    ecx
2056 00006162 F3A5                <1>       rep    movsd  ; shift up (to the head)
2057                              <1> swpqs_5:
2058 00006164 31C0                <1>       xor    eax, eax
2059 00006166 8907                <1>       mov    [edi], eax
2060                              <1> swpqs_6:
2061 00006168 59                  <1>       pop    ecx
2062 00006169 5E                  <1>       pop    esi
2063 0000616A 5F                  <1>       pop    edi
2064                              <1> swpqs_retn:
2065 0000616B C3                  <1>       retn
2066                              <1> swpqs_7:
2067 0000616C 89F7                <1>       mov    edi, esi ; head
2068 0000616E AD                  <1>       lodsd
2069                              <1>       ; 20/07/2015
2070 0000616F 89C3                <1>       mov    ebx, eax
2071 00006171 81E300F0FFFF        <1>       and    ebx, ~PAGE_OFF ; ~0FFFh
2072                              <1>                     ; ebx = virtual address (at page boundary)
2073 00006177 25FF0F0000          <1>       and    eax, PAGE_OFF ; 0FFFh
2074                              <1>                     ; ax = process number (1 to 4095)
2075 0000617C 3A05[B3030300]      <1>       cmp    al, [u.uno]
2076                              <1>              ; Max. 16 (nproc) processes for Retro UNIX 386 v1
2077 00006182 7507                <1>       jne    short swpqs_8
2078 00006184 A1[B8030300]        <1>       mov    eax, [u.pgdir]
2079 00006189 EB28                <1>       jmp    short swpqs_9
2080                              <1> swpqs_8:
2081                              <1>       ; 09/06/2016
2082 0000618B 80B8[AF000300]00    <1>       cmp    byte [eax+p.stat-1], 0
2083 00006192 76C4                <1>       jna    short swpqs_3     ; free (or terminated) process
2084 00006194 80B8[AF000300]02    <1>       cmp    byte [eax+p.stat-1], 2 ; waiting
2085 0000619B 77BB                <1>       ja     short swpqs_3         ; zombie (3) or undefined ?
2086                              <1>
2087                              <1>       ;shl   ax, 2
2088 0000619D C0E002              <1>       shl    al, 2
2089 000061A0 8B80[BC000300]      <1>       mov    eax, [eax+p.upage-4]
2090 000061A6 09C0                <1>       or     eax, eax
2091 000061A8 74AE                <1>       jz     short swpqs_3 ; invalid upage
2092 000061AA 83C05C              <1>       add    eax, u.pgdir - user
2093                              <1>                     ; u.pgdir value for the process
2094                              <1>                     ; is in [eax]
2095 000061AD 8B00                <1>       mov    eax, [eax]
2096 000061AF 21C0                <1>       and    eax, eax
2097 000061B1 74A5                <1>       jz     short swpqs_3 ; invalid page directory
2098                              <1> swpqs_9:
2099 000061B3 52                  <1>       push   edx
2100                              <1>       ; eax = page directory
2101                              <1>       ; ebx = virtual address
2102 000061B4 E82BFBFFFF          <1>       call   get_pte
2103 000061B9 89D3                <1>       mov    ebx, edx     ; PTE address
2104 000061BB 5A                  <1>       pop    edx
2105                              <1>       ; 10/06/2016
2106 000061BC 723A                <1>       jc     short swpqs_13 ; empty PDE
2107                              <1>       ; EAX = PTE value
2108 000061BE A801                <1>       test   al, PTE_A_PRESENT ; bit 0 = 1
2109 000061C0 7436                <1>       jz     short swpqs_13  ; Drop non-present page
2110                              <1>                              ; from the queue (head)
2111 000061C2 A802                <1>       test   al, PTE_A_WRITE     ; bit 1 = 0 (read only)
2112 000061C4 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 000061C6 0FBAF005           <1>         btr     eax, PTE_A_ACCESS_BIT ; reset 'accessed' bit
2118 000061CA 7210                <1>       jc     short swpqs_11  ; accessed page
2119                              <1>
2120 000061CC 49                  <1>       dec    ecx
2121 000061CD 66890D[60050300]    <1>       mov    [swpq_count], cx
2122 000061D4 7402                <1>         jz      short swpqs_10
2123                              <1>             ; esi = head + 4
2124                              <1>             ; edi = head
2125 000061D6 F3A5                <1>       rep    movsd  ; n = 1 to k-1, [n - 1] = [n]
2126                              <1> swpqs_10:
2127 000061D8 890F                <1>       mov    [edi], ecx ; 0
2128 000061DA EB8C                <1>       jmp    short swpqs_6 ; 26/03/2017
2129                              <1>
2130                              <1> swpqs_11:
2131 000061DC 8903                <1>       mov    [ebx], eax     ; save changed attribute
2132                              <1>       ; Rotation (head -> tail)
2133 000061DE 49                  <1>       dec    ecx     ; entry count -> last entry number
2134 000061DF 74F7                <1>       jz     short swpqs_10
2135                              <1>             ; esi = head + 4
2136                              <1>             ; edi = head
2137 000061E1 8B07                <1>       mov    eax, [edi] ; 20/07/2015
2138 000061E3 F3A5                <1>       rep    movsd  ; n = 1 to k-1, [n - 1] = [n]
2139 000061E5 8907                <1>       mov    [edi], eax ; head -> tail ; [k] = [1]
2140                              <1>
2141 000061E7 668B0D[60050300]    <1>       mov    cx, [swpq_count]
2142                              <1>
2143                              <1> swpqs_12:
2144 000061EE BE00E00800          <1>       mov    esi, swap_queue ; head
2145 000061F3 E974FFFFFF          <1>         jmp     swpqs_7
2146                              <1>
```

```
2147                                <1> swpqs_13:
2148 000061F8 49                   <1>         dec     ecx
2149 000061F9 66890D[60050300]     <1>         mov     [swpq_count], cx
2150 00006200 0F845EFFFFFF         <1>           jz      swpqs_5
2151 00006206 EBE6                 <1>         jmp     short swpqs_12
2152                                <1>
2153                                <1> add_to_swap_queue:
2154                                <1> ; temporary - 16/09/2015
2155 00006208 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 00006209 53                   <1>         push    ebx
2179 0000620A 6681E300F0           <1>           and     bx, ~PAGE_OFF ; ~0FFFh ; reset bits, 0 to 11
2180 0000620F 8A1D[B3030300]       <1>         mov     bl, [u.uno] ; current process number
2181 00006215 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 0000621A 0FB705[60050300]     <1>         movzx   eax, word [swpq_count]
2185 00006221 663D0004             <1>         cmp     ax, 1024
2186 00006225 7205                 <1>         jb      short atsq_1
2187 00006227 6629C0               <1>         sub     ax, ax
2188 0000622A 5B                   <1>         pop     ebx
2189 0000622B C3                   <1>         retn
2190                                <1> atsq_1:
2191 0000622C 56                   <1>         push    esi
2192 0000622D BE00E00800           <1>         mov     esi, swap_queue
2193 00006232 6621C0               <1>         and     ax, ax
2194 00006235 740A                 <1>         jz      short atsq_2
2195 00006237 66C1E002             <1>         shl     ax, 2  ; convert to offset
2196 0000623B 01C6                 <1>         add     esi, eax
2197 0000623D 66C1E802             <1>         shr     ax, 2
2198                                <1> atsq_2:
2199 00006241 6640                 <1>         inc     ax
2200 00006243 891E                 <1>         mov     [esi], ebx ; Virtual address + [u.uno] combination
2201 00006245 66A3[60050300]       <1>         mov     [swpq_count], ax
2202 0000624B 5E                   <1>         pop     esi
2203 0000624C 5B                   <1>         pop     ebx
2204 0000624D 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 0000624E 53                   <1>         push    ebx
2222 0000624F 52                   <1>         push    edx
2223                                <1>         ;
2224 00006250 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 00006253 89C2                 <1>         mov     edx, eax
2228                                <1>         ; 15/09/2015
2229 00006255 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 00006258 80E2FC               <1>         and     dl, 0FCh             ; clear lower 2 bits
2233                                <1>                                      ; (to get 32 bit position)
2234                                <1>         ;
2235 0000625B BB00000D00           <1>         mov     ebx, swap_alloc_table ; Swap Allocation Table address
2236 00006260 01D3                 <1>         add     ebx, edx
2237 00006262 83E01F               <1>         and     eax, 1Fh             ; lower 5 bits only
2238                                <1>                                      ; (allocation bit position)
2239 00006265 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 0000626B 7305                 <1>         jnb     short uswpbl_1           ; no
2243 0000626D A3[6E050300]         <1>         mov     [swpd_next], eax      ; yes
2244                                <1> uswpbl_1:
2245 00006272 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 00006275 F5                   <1>         cmc                          ; complement carry flag
2249 00006276 7206                 <1>         jc      short uswpbl_2           ; do not increase swfd_free count
2250                                <1>                                      ; if the block is already deallocated
2251                                <1>                                      ; before.
```

```
2252 00006278 FF05[6A050300]      <1>         inc     dword [swpd_free]
2253                              <1> uswpbl_2:
2254 0000627E 5A                  <1>         pop     edx
2255 0000627F 5B                  <1>         pop     ebx
2256 00006280 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 00006281 53                  <1>         push   ebx
2281 00006282 51                  <1>         push   ecx
2282                              <1>         ;
2283 00006283 BB00000D00          <1>         mov    ebx, swap_alloc_table ; Swap Allocation Table offset
2284 00006288 89D9                <1>         mov    ecx, ebx
2285 0000628A 031D[6E050300]      <1>         add    ebx, [swpd_next] ; Free block searching starts from here
2286                              <1>                                 ; next_free_swap_block >> 5
2287 00006290 030D[72050300]      <1>         add    ecx, [swpd_last] ; Free block searching ends here
2288                              <1>                                 ; (total_swap_blocks - 1) >> 5
2289                              <1> lswbl_scan:
2290 00006296 39CB                <1>         cmp    ebx, ecx
2291 00006298 770A                <1>         ja     short lswbl_notfound
2292                              <1>         ;
2293 0000629A 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 0000629D 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 0000629F 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 000062A2 EBF2                <1>         jmp           short lswbl_scan
2310                              <1>         ;
2311                              <1> lswbl_notfound:
2312 000062A4 81E900000D00        <1>         sub    ecx, swap_alloc_table
2313 000062AA 890D[6E050300]      <1>         mov    [swpd_next], ecx ; next/first free page = last page
2314                              <1>                                 ; (unlink_swap_block procedure will change it)
2315 000062B0 31C0                <1>         xor    eax, eax
2316 000062B2 A3[6A050300]        <1>         mov    [swpd_free], eax
2317 000062B7 F9                  <1>         stc
2318                              <1> lswbl_ok:
2319 000062B8 59                  <1>         pop    ecx
2320 000062B9 5B                  <1>         pop    ebx
2321 000062BA C3                  <1>         retn
2322                              <1>         ;
2323                              <1> ;out_of_swpspc:
2324                              <1> ;       stc
2325                              <1> ;       retn
2326                              <1>
2327                              <1> lswbl_found:
2328 000062BB 89D9                <1>         mov    ecx, ebx
2329 000062BD 81E900000D00        <1>         sub    ecx, swap_alloc_table
2330 000062C3 890D[6E050300]      <1>         mov    [swpd_next], ecx ; Set first free block searching start
2331                              <1>                                 ; address/offset (to the next)
2332 000062C9 FF0D[6A050300]      <1>           dec     dword [swpd_free] ; 1 block has been allocated (X = X-1)
2333                              <1>         ;
2334 000062CF 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 000062D2 C1E105              <1>         shl    ecx, 5         ; (block offset * 32) + block index
2341 000062D5 01C8                <1>         add    eax, ecx       ; = block number
2342 000062D7 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 000062DA 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 ->
```

```
                                <1>       ;      ESI = Logical disk description table address
                                <1>       ;      EBX = Memory page (buffer) address (physical!)
                                <1>       ;      EAX = Sector adress (offset address, logical sector number)
                                <1>       ;      ECX = Sector count
                                <1>       ;
                                <1>       ;
000062DC C3                     <1>       retn
                                <1>
                                <1> logical_disk_write:
                                <1>       ; 20/07/2015
                                <1>       ; 09/03/2015 (temporary code here)
                                <1>       ;
                                <1>       ; INPUT ->
                                <1>       ;      ESI = Logical disk description table address
                                <1>       ;      EBX = Memory page (buffer) address (physical!)
                                <1>       ;      EAX = Sector adress (offset address, logical sector number)
                                <1>       ;      ECX = Sector count
                                <1>       ;
000062DD C3                     <1>       retn
                                <1>
                                <1> get_physical_addr:
                                <1>       ; 26/03/2017
                                <1>       ; 20/02/2017
                                <1>       ; 27/05/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>       ; 18/10/2015
                                <1>       ; 29/07/2015
                                <1>       ; 20/07/2015
                                <1>       ; 04/06/2015
                                <1>       ; 20/05/2015
                                <1>       ; 28/04/2015
                                <1>       ; 18/04/2015
                                <1>       ; Get physical address
                                <1>       ;      (allocates a new page for user if it is not present)
                                <1>       ;
                                <1>       ; (This subroutine is needed for mapping user's virtual
                                <1>       ; (buffer) address to physical address (of the buffer).)
                                <1>       ; ('sys write', 'sys read' system calls...)
                                <1>       ;
                                <1>       ; INPUT ->
                                <1>       ;      EBX = virtual address
                                <1>       ;      u.pgdir = page directory (physical) address
                                <1>       ;
                                <1>       ; OUTPUT ->
                                <1>       ;      EAX = physical address
                                <1>       ;      EBX = linear address
                                <1>       ;      EDX = physical address of the page frame
                                <1>       ;            (with attribute bits)
                                <1>       ;      ECX = byte count within the page frame
                                <1>       ;
                                <1>       ; Modified Registers -> EAX, EBX, ECX, EDX
                                <1>       ;
000062DE 81C300004000           <1>       add    ebx, CORE ; 18/10/2015
                                <1> get_physical_addr_x: ; 27/05/2016
000062E4 A1[B8030300]           <1>       mov    eax, [u.pgdir]
000062E9 E8F6F9FFFF             <1>       call   get_pte
                                <1>              ; EDX = Page table entry address (if CF=0)
                                <1>                ;       Page directory entry address (if CF=1)
                                <1>              ;       (Bit 0 value is 0 if PT is not present)
                                <1>              ; EAX = Page table entry value (page address)
                                <1>              ;      CF = 1 -> PDE not present or invalid ?
000062EE 731C                   <1>       jnc    short gpa_1
                                <1>       ;
000062F0 E8D4F8FFFF             <1>       call   allocate_page
000062F5 7248                   <1>       jc     short gpa_im_err  ; 'insufficient memory' error
                                <1> gpa_0:
000062F7 E847F9FFFF             <1>       call   clear_page
                                <1>       ; EAX = Physical (base) address of the allocated (new) page
000062FC 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)
000062FE 8902                   <1>       mov    [edx], eax ; Let's put the new page directory entry here !
00006300 A1[B8030300]           <1>       mov    eax, [u.pgdir]
00006305 E8DAF9FFFF             <1>       call   get_pte
0000630A 7233                   <1>       jc     short gpa_im_err ; 'insufficient memory' error
                                <1> gpa_1:
                                <1>       ; EAX = PTE value, EDX = PTE address
0000630C A801                   <1>       test   al, PTE_A_PRESENT
0000630E 751F                   <1>       jnz    short gpa_3 ; 26/03/2017
00006310 09C0                   <1>       or     eax, eax
00006312 7456                   <1>       jz     short gpa_7  ; Allocate a new page
                                <1>       ; 20/07/2015
00006314 55                     <1>       push   ebp
00006315 89DD                   <1>       mov    ebp, ebx ; virtual (linear) address
                                <1>       ; reload swapped page
00006317 E878000000             <1>       call   reload_page ; 28/04/2015
0000631C 5D                     <1>       pop    ebp
0000631D 724A                   <1>       jc     short gpa_retn
                                <1> gpa_2:
                                <1>       ; 26/03/2017
                                <1>       ; 20/02/2017
                                <1>       ; If a page will contain a Signal Response Byte
                                <1>       ; it must not be swapped out, because
                                <1>       ; timer service or irq callback service
                                <1>       ; will write a signal return/response byte
                                <1>       ; directly by using physical address of Signal
                                <1>       ; Response Byte.(Even if process is not running,
                                <1>       ; or it is running with swapped out pages.)
                                <1>       ;
                                <1>       ; 'no_page_swap' will be set by 'systimer' or
                                <1>       ; 'syscalbac' sistem functions/calls. (*)
                                <1>       ;
0000631F 803D[62930100]00       <1>       cmp    byte [no_page_swap], 0
00006326 761D                   <1>       jna    short gpa_4 ; this page can be swapped out
                                <1>       ; this page must not be swapped out
                                <1>       ; but 'no_page_swap' must be reset here
```

```
2462                                <1>       ; imediately for other callers (*)
2463                                <1>       ; (otherwise, swap queue would not be long enough)
2464 00006328 E84B000000          <1>       call   gpa_8 ; 26/03/2017
2465 0000632D EB1D                <1>       jmp    short gpa_5
2466                                <1> gpa_3:
2467                                <1>       ; 26/03/2017
2468 0000632F 803D[62930100]00    <1>       cmp    byte [no_page_swap], 0
2469 00006336 7618                <1>       jna    short gpa_6 ; this page can be swapped out
2470 00006338 E83B000000          <1>       call   gpa_8
2471 0000633D EB11                <1>       jmp    short gpa_6
2472                                <1>
2473                                <1> gpa_im_err:
2474 0000633F B804000000          <1>       mov    eax, ERR_MINOR_IM ; Insufficient memory (minor) error!
2475                                <1>                                ; Major error = 0 (No protection fault)
2476 00006344 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 00006345 50                  <1>       push   eax
2482                                <1>       ; EBX = Linear (CORE+virtual) address ; 20/02/2017
2483 00006346 E8BDFEFFFF          <1>       call   add_to_swap_queue
2484 0000634B 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 0000634C 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 0000634E 8902                <1>       mov    [edx], eax  ; Update PTE value
2494                                <1> gpa_6:
2495                                <1>       ; 18/10/2015
2496 00006350 89D9                <1>       mov    ecx, ebx
2497 00006352 81E1FF0F0000        <1>       and    ecx, PAGE_OFF
2498 00006358 89C2                <1>       mov    edx, eax
2499 0000635A 662500F0            <1>       and    ax, PTE_A_CLEAR
2500 0000635E 01C8                <1>       add    eax, ecx
2501 00006360 F7D9                <1>       neg    ecx ; 1 -> -1 (0FFFFFFFFh), 4095 (0FFFh) -> -4095
2502 00006362 81C100100000        <1>       add    ecx, PAGE_SIZE
2503 00006368 F8                  <1>       clc
2504                                <1> gpa_retn:
2505 00006369 C3                  <1>       retn
2506                                <1> gpa_7:
2507 0000636A E85AF8FFFF          <1>       call   allocate_page
2508 0000636F 72CE                <1>       jc     short gpa_im_err ; 'insufficient memory' error
2509 00006371 E8CDF8FFFF          <1>       call   clear_page
2510 00006376 EBA7                <1>       jmp    short gpa_2
2511                                <1>
2512                                <1> gpa_8: ; 26/03/2017
2513 00006378 C605[62930100]00    <1>       mov    byte [no_page_swap], 0
2514 0000637F 53                  <1>       push   ebx
2515 00006380 50                  <1>       push   eax ; 26/03/2017
2516 00006381 6681E300F0          <1>         and     bx, ~PAGE_OFF ; ~0FFFh ; reset bits, 0 to 11
2517 00006386 8A1D[B3030300]      <1>       mov    bl, [u.uno] ; current process number
2518 0000638C E89DFDFFFF          <1>       call   swap_queue_shift ; drop from the queue if
2519                                <1>                               ; it is already on the queue
2520 00006391 58                  <1>       pop    eax ; 26/03/2017
2521 00006392 5B                  <1>       pop    ebx
2522 00006393 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 00006394 D1E8                <1>       shr    eax, 1   ; Convert PTE value to swap disk address
2542 00006396 53                  <1>       push   ebx      ;
2543 00006397 89C3                <1>       mov    ebx, eax ; Swap disk (offset) address
2544 00006399 E82BF8FFFF          <1>       call   allocate_page
2545 0000639E 720C                <1>       jc     short rlp_im_err
2546 000063A0 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 000063A1 E862FCFFFF          <1>       call   swap_in
2551 000063A6 720B                <1>       jc     short rlp_swp_err  ; (swap disk/file read error)
2552 000063A8 89D8                <1>       mov    eax, ebx
2553                                <1> rlp_retn:
2554 000063AA 5B                  <1>       pop    ebx
2555 000063AB C3                  <1>       retn
2556                                <1>
2557                                <1> rlp_im_err:
2558 000063AC B804000000          <1>       mov    eax, ERR_MINOR_IM ; Insufficient memory (minor) error!
2559                                <1>                                ; Major error = 0 (No protection fault)
2560 000063B1 EBF7                <1>       jmp    short rlp_retn
2561                                <1>
2562                                <1> rlp_swp_err:
2563 000063B3 B828000000          <1>       mov    eax, SWP_DISK_READ_ERR ; Swap disk read error !
2564 000063B8 EBF0                <1>       jmp    short rlp_retn
2565                                <1>
2566                                <1>
```

```
                                <1> copy_page_dir:
                                <1>       ; 19/09/2015
                                <1>       ; temporary - 07/09/2015
                                <1>       ; 07/09/2015 (Retro UNIX 386 v1 - beginning)
                                <1>       ;
                                <1>       ; INPUT ->
                                <1>       ;     [u.pgdir] = PHYSICAL (real/flat) ADDRESS of the parent's
                                <1>       ;                 page directory.
                                <1>       ; OUTPUT ->
                                <1>       ;     EAX =  PHYSICAL (real/flat) ADDRESS of the child's
                                <1>       ;            page directory.
                                <1>       ;     (New page directory with new page table entries.)
                                <1>       ;     (New page tables with read only copies of the parent's
                                <1>       ;     pages.)
                                <1>       ;     EAX = 0 -> Error (CF = 1)
                                <1>       ;
                                <1>       ; Modified Registers -> none (except EAX)
                                <1>       ;
000063BA E80AF8FFFF             <1>       call   allocate_page
000063BF 723E                   <1>       jc     short cpd_err
                                <1>       ;
000063C1 55                     <1>       push   ebp ; 20/07/2015
000063C2 56                     <1>       push   esi
000063C3 57                     <1>       push   edi
000063C4 53                     <1>       push   ebx
000063C5 51                     <1>       push   ecx
000063C6 8B35[B8030300]         <1>       mov    esi, [u.pgdir]
000063CC 89C7                   <1>       mov    edi, eax
000063CE 50                     <1>       push   eax ; save child's page directory address
                                <1>       ; copy PDE 0 from the parent's page dir to the child's page dir
                                <1>       ; (use same system space for all user page tables)
000063CF A5                     <1>       movsd
000063D0 BD00004000             <1>       mov    ebp, 1024*4096 ; pass the 1st 4MB (system space)
000063D5 B9FF030000             <1>       mov    ecx, (PAGE_SIZE / 4) - 1 ; 1023
                                <1> cpd_0:
000063DA AD                     <1>       lodsd
                                <1>       ;or    eax, eax
                                <1>         ;jnz     short cpd_1
000063DB A801                   <1>       test   al, PDE_A_PRESENT ;  bit 0 =  1
000063DD 7508                   <1>       jnz    short cpd_1
                                <1>       ; (virtual address at the end of the page table)
000063DF 81C500004000           <1>       add    ebp, 1024*4096 ; page size * PTE count
000063E5 EB0F                   <1>       jmp    short cpd_2
                                <1> cpd_1:
000063E7 662500F0               <1>       and    ax, PDE_A_CLEAR ; 0F000h ; clear attribute bits
000063EB 89C3                   <1>       mov    ebx, eax
                                <1>       ; EBX = Parent's page table address
000063ED E81F000000             <1>       call   copy_page_table
000063F2 720C                   <1>       jc     short cpd_p_err
                                <1>       ; EAX = Child's page table address
000063F4 0C07                   <1>       or     al, PDE_A_PRESENT + PDE_A_WRITE + PDE_A_USER
                                <1>                ; set bit 0, bit 1 and bit 2 to 1
                                <1>                ; (present, writable, user)
                                <1> cpd_2:
000063F6 AB                     <1>       stosd
000063F7 E2E1                   <1>       loop   cpd_0
                                <1>       ;
000063F9 58                     <1>       pop    eax  ; restore child's page directory address
                                <1> cpd_3:
000063FA 59                     <1>       pop    ecx
000063FB 5B                     <1>       pop    ebx
000063FC 5F                     <1>       pop    edi
000063FD 5E                     <1>       pop    esi
000063FE 5D                     <1>       pop    ebp
                                <1> cpd_err:
000063FF C3                     <1>       retn
                                <1> cpd_p_err:
                                <1>       ; release the allocated pages missing (recover free space)
00006400 58                     <1>       pop    eax  ; the new page directory address (physical)
00006401 8B1D[B8030300]         <1>       mov    ebx, [u.pgdir] ; parent's page directory address
00006407 E8F6F8FFFF             <1>       call   deallocate_page_dir
0000640C 29C0                   <1>       sub    eax, eax ; 0
0000640E F9                     <1>       stc
0000640F EBE9                   <1>       jmp    short cpd_3
                                <1>
                                <1> copy_page_table:
                                <1>       ; 19/09/2015
                                <1>       ; temporary - 07/09/2015
                                <1>       ; 07/09/2015 (Retro UNIX 386 v1 - beginning)
                                <1>       ;
                                <1>       ; INPUT ->
                                <1>       ;     EBX = PHYSICAL (real/flat) ADDRESS of the parent's page table.
                                <1>       ;     EBP = page table entry index (from 'copy_page_dir')
                                <1>       ; OUTPUT ->
                                <1>       ;     EAX = PHYSICAL (real/flat) ADDRESS of the child's page table.
                                <1>       ;     EBP = (recent) page table index (for 'add_to_swap_queue')
                                <1>       ;     CF = 1 -> error
                                <1>       ;
                                <1>       ; Modified Registers -> EBP (except EAX)
                                <1>       ;
00006411 E8B3F7FFFF             <1>       call   allocate_page
00006416 725A                   <1>       jc     short cpt_err
                                <1>       ;
00006418 50                     <1>       push   eax ; *
                                <1>       ;push  ebx
00006419 56                     <1>       push   esi
0000641A 57                     <1>       push   edi
0000641B 52                     <1>       push   edx
0000641C 51                     <1>       push   ecx
                                <1>       ;
0000641D 89DE                   <1>       mov    esi, ebx
0000641F 89C7                   <1>       mov    edi, eax
00006421 89C2                   <1>       mov    edx, eax
00006423 81C200100000           <1>       add    edx, PAGE_SIZE
                                <1> cpt_0:
```

```
2672 00006429 AD                <1>       lodsd
2673 0000642A A801              <1>       test   al, PTE_A_PRESENT ;  bit 0 =  1
2674 0000642C 750B              <1>       jnz    short cpt_1
2675 0000642E 21C0              <1>       and    eax, eax
2676 00006430 7430              <1>       jz     short cpt_2
2677                            <1>       ; ebp = virtual (linear) address of the memory page
2678 00006432 E85DFFFFFF        <1>       call   reload_page ; 28/04/2015
2679 00006437 7234              <1>       jc     short cpt_p_err
2680                            <1> cpt_1:
2681 00006439 662500F0          <1>       and    ax, PTE_A_CLEAR ; 0F000h ; clear attribute bits
2682 0000643D 89C1              <1>       mov    ecx, eax
2683                            <1>       ; Allocate a new page for the child process
2684 0000643F E885F7FFFF        <1>       call   allocate_page
2685 00006444 7227              <1>       jc     short cpt_p_err
2686 00006446 57                <1>       push   edi
2687 00006447 56                <1>       push   esi
2688 00006448 89CE              <1>       mov    esi, ecx
2689 0000644A 89C7              <1>       mov    edi, eax
2690 0000644C B900040000        <1>       mov    ecx, PAGE_SIZE/4
2691 00006451 F3A5              <1>       rep    movsd  ; copy page (4096 bytes)
2692 00006453 5E                <1>       pop    esi
2693 00006454 5F                <1>       pop    edi
2694                            <1>       ;
2695 00006455 53                <1>       push   ebx
2696 00006456 50                <1>       push   eax
2697 00006457 89EB              <1>       mov    ebx, ebp
2698                            <1>       ; ebx = virtual address of the memory page
2699 00006459 E8AAFDFFFF        <1>       call   add_to_swap_queue
2700 0000645E 58                <1>       pop    eax
2701 0000645F 5B                <1>       pop    ebx
2702                            <1>       ;
2703                            <1>       ;or    ax, PTE_A_USER+PTE_A_PRESENT
2704 00006460 0C07              <1>       or     al, PTE_A_USER+PTE_A_WRITE+PTE_A_PRESENT
2705                            <1> cpt_2:
2706 00006462 AB                <1>       stosd  ; EDI points to child's PTE
2707                            <1>       ;
2708 00006463 81C500100000      <1>       add    ebp, 4096 ; 20/07/2015 (next page)
2709                            <1>       ;
2710 00006469 39D7              <1>       cmp    edi, edx
2711 0000646B 72BC              <1>       jb     short cpt_0
2712                            <1> cpt_p_err:
2713 0000646D 59                <1>       pop    ecx
2714 0000646E 5A                <1>       pop    edx
2715 0000646F 5F                <1>       pop    edi
2716 00006470 5E                <1>       pop    esi
2717                            <1>       ;pop   ebx
2718 00006471 58                <1>       pop    eax ; *
2719                            <1> cpt_err:
2720 00006472 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
2760                            <1>       ; 'deallocate_memory_block' procedure.
2761                            <1>
2762 00006473 52                <1>       push   edx ; *
2763 00006474 BAFF0F0000        <1>       mov    edx, PAGE_SIZE - 1   ; 4095
2764 00006479 01D0              <1>       add    eax, edx
2765 0000647B 01D1              <1>       add    ecx, edx
2766 0000647D C1E90C            <1>       shr    ecx, PAGE_SHIFT          ; 12
2767                            <1>
2768                            <1>       ; ECX = number of contiguous pages to be allocated
2769 00006480 8B15[98810100]    <1>       mov    edx, [free_pages]
2770                            <1>       ; 01/05/2017
2771                            <1>       ;or    ecx, ecx
2772                            <1>       ;jz    short amb3
2773                            <1>       ; If ECX=0, set cf to 1 and return with max. available mem block size
2774                            <1>
2775 00006486 39D1              <1>       cmp    ecx, edx
2776 00006488 7760              <1>       ja     short amb_3
```

```
2777                                <1>
2778 0000648A C1E80C               <1>       shr    eax, PAGE_SHIFT      ; 12
2779                                <1>
2780 0000648D 89C2                 <1>       mov    edx, eax           ; page number
2781 0000648F C1EA03               <1>       shr    edx, 3             ; to get offset to M.A.T.
2782                                <1>                                 ; (1 allocation bit = 1 page)
2783                                <1>                                 ; (1 allocation bytes = 8 pages)
2784 00006492 80E2FC               <1>       and    dl, 0FCh           ; clear lower 2 bits
2785                                <1>                                 ; (to get 32 bit position)
2786 00006495 53                   <1>       push   ebx ; **
2787                                <1> amb_0:
2788 00006496 890D[4C8D0100]       <1>       mov    [mem_ipg_count], ecx ; initial (reset) value of page count
2789 0000649C 890D[508D0100]       <1>       mov    [mem_pg_count], ecx
2790 000064A2 31C9                 <1>       xor    ecx, ecx ; 0
2791 000064A4 890D[548D0100]       <1>       mov    [mem_aperture], ecx ; 0
2792 000064AA 890D[588D0100]       <1>       mov    [mem_max_aperture], ecx ; 0
2793                                <1>
2794 000064B0 BB00001000           <1>       mov    ebx, MEM_ALLOC_TBL   ; Memory Allocation Table address.
2795 000064B5 3B15[9C810100]       <1>       cmp    edx, [next_page]     ; Is the beginning page address lower
2796                                <1>                                 ; than the address in 'next_page' ?
2797                                <1>                                 ; (the first/next free page of user space)
2798 000064BB 7208                 <1>       jb     short amb_1
2799 000064BD 3B15[A0810100]       <1>       cmp    edx, [last_page]     ; is the beginning page address higher
2800                                <1>                                 ; than the address in 'last_page' ?
2801                                <1>                                 ; (end of the memory)
2802 000064C3 7606                 <1>       jna    short amb_2        ; no
2803                                <1> amb_1:
2804 000064C5 8B15[9C810100]       <1>       mov    edx, [next_page]     ; M.A.T. offset (1 M.A.T. byte = 8 pages)
2805                                <1> amb_2:
2806 000064CB 01D3                 <1>       add    ebx, edx
2807                                <1>
2808                                <1>       ; 28/04/2017
2809                                <1>       ;xor   ecx, ecx
2810 000064CD 0FBC0B               <1>       bsf    ecx, [ebx]         ; 0 to 31
2811 000064D0 89D0                 <1>       mov    eax, edx
2812 000064D2 C1E003               <1>       shl    eax, 3             ; *8
2813 000064D5 01C8                 <1>       add    eax, ecx           ; beginning page number
2814                                <1>
2815 000064D7 A3[5C8D0100]         <1>       mov    [mem_pg_pos], eax    ; beginning page no (for curr. mem. aperture)
2816 000064DC A3[608D0100]         <1>       mov    [mem_max_pg_pos], eax ; beginning page no for max. mem. aperture
2817                                <1>
2818 000064E1 83E01F               <1>       and    eax, 1Fh           ; lower 5 bits only (0 to 31)
2819                                <1>                                 ; (allocation bit position)
2820 000064E4 750E                 <1>       jnz    short amb_4        ; 0
2821 000064E6 B120                 <1>       mov    cl, 32
2822 000064E8 EB4B                 <1>       jmp    short amb_10
2823                                <1>
2824                                <1> amb_3:       ; out_of_memory
2825 000064EA 31C0                 <1>       xor    eax, eax ; 0
2826 000064EC 89D1                 <1>       mov    ecx, edx ; free pages
2827 000064EE C1E10C               <1>       shl    ecx, PAGE_SHIFT
2828 000064F1 5A                   <1>       pop    edx ; *
2829 000064F2 F9                   <1>       stc
2830 000064F3 C3                   <1>       retn
2831                                <1> amb_4:
2832 000064F4 8B13                 <1>       mov    edx, [ebx]
2833 000064F6 88C1                 <1>       mov    cl, al ; 1 to 31
2834 000064F8 D3EA                 <1>       shr    edx, cl
2835 000064FA 89D0                 <1>       mov    eax, edx
2836                                <1> amb_5:
2837 000064FC D1E8                 <1>       shr    eax, 1 ; (***)
2838 000064FE 7317                 <1>       jnc    short amb_7
2839 00006500 FF05[548D0100]       <1>       inc    dword [mem_aperture]
2840 00006506 FF0D[508D0100]       <1>       dec    dword [mem_pg_count]
2841 0000650C 7470                 <1>       jz     short amb_15
2842                                <1> amb_6:
2843                                <1>       ; 28/04/2017
2844 0000650E FEC1                 <1>       inc    cl
2845 00006510 80F920               <1>       cmp    cl, 32
2846 00006513 730D                 <1>       jnb    short amb_9
2847 00006515 EBE5                 <1>       jmp    short amb_5
2848                                <1> amb_7:
2849 00006517 50                   <1>       push   eax ; (***) allocation bits (in shifted status)
2850 00006518 E81B010000           <1>       call   amb_26 ; set maximum memory aperture (free memory block size)
2851 0000651D 58                   <1>       pop    eax ; (***)
2852 0000651E EBEE                 <1>       jmp    short amb_6
2853                                <1> amb_8:
2854                                <1>       ; 28/04/2017
2855 00006520 B120                 <1>       mov    cl, 32
2856                                <1> amb_9:
2857 00006522 89DA                 <1>       mov    edx, ebx
2858 00006524 81EA00001000         <1>       sub    edx, MEM_ALLOC_TBL
2859 0000652A 3B15[A0810100]       <1>       cmp    edx, [last_page]
2860 00006530 7336                 <1>       jnb    short amb_14 ; contiguous pages not enough
2861 00006532 83C304               <1>       add    ebx, 4
2862                                <1> amb_10:
2863 00006535 8B03                 <1>       mov    eax, [ebx]
2864 00006537 21C0                 <1>       and    eax, eax
2865 00006539 7408                 <1>         jz     short amb_11 ; there is not a free page bit in this alloc dword
2866 0000653B 40                   <1>       inc    eax ; 0FFFFFFFFh -> 0
2867 0000653C 740C                 <1>       jz     short amb_12 ; all of bits are set (32 free pages)
2868 0000653E 48                   <1>       dec    eax
2869 0000653F 28C9                 <1>       sub    cl, cl ; 0
2870 00006541 EBB9                 <1>       jmp    short amb_5
2871                                <1> amb_11:
2872 00006543 E8F0000000           <1>       call   amb_26 ; set maximum memory aperture (free memory block size)
2873 00006548 EBD8                 <1>       jmp    short amb_9
2874                                <1> amb_12:
2875 0000654A 390D[508D0100]       <1>       cmp    [mem_pg_count], ecx ; 32
2876 00006550 7306                 <1>       jnb    short amb_13
2877 00006552 8B0D[508D0100]       <1>       mov    ecx, [mem_pg_count]
2878                                <1> amb_13:
2879 00006558 010D[548D0100]       <1>       add    [mem_aperture], ecx
2880 0000655E 290D[508D0100]       <1>       sub    [mem_pg_count], ecx
2881 00006564 7618                 <1>       jna    short amb_15
```

```
2882 00006566 EBBA              <1>       jmp    short amb_9 ; 01/05/2017
2883                            <1> amb_14:
2884 00006568 E8CB000000        <1>       call   amb_26 ; 28/04/2017
2885 0000656D A1[608D0100]      <1>       mov    eax, [mem_max_pg_pos] ; begin address of max. mem aperture
2886 00006572 8B0D[588D0100]    <1>       mov    ecx, [mem_max_aperture] ; max. (largest) memory aperture
2887 00006578 F9                <1>       stc
2888 00006579 E9AF000000        <1>         jmp     amb_25
2889                            <1>
2890                            <1> amb_15: ; OK !
2891 0000657E A1[5C8D0100]      <1>       mov    eax, [mem_pg_pos]    ; Beginning address as page number
2892 00006583 8B0D[548D0100]    <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 00006589 89C2              <1>       mov    edx, eax
2896                            <1>       ; 25/04/2017
2897 0000658B C1EA03            <1>       shr    edx, 3        ; 8 pages in one allocation byte
2898 0000658E 80E2FC            <1>       and    dl, 0FCh      ; clear lower 2 bits
2899                            <1>                            ; (for dword/32bit positioning)
2900                            <1>
2901 00006591 BB00001000        <1>       mov    ebx, MEM_ALLOC_TBL
2902 00006596 01D3              <1>       add    ebx, edx
2903 00006598 83E01F            <1>       and    eax, 1Fh ; 31
2904                            <1>       ; 03/04/2016
2905 0000659B BA20000000        <1>       mov    edx, 32
2906 000065A0 28C2              <1>       sub    dl, al
2907 000065A2 39CA              <1>       cmp    edx, ecx      ; ecx >= 1
2908 000065A4 7602              <1>       jna    short amb_17
2909 000065A6 89CA              <1>       mov    edx, ecx
2910                            <1> amb_17:
2911 000065A8 29D1              <1>       sub    ecx, edx
2912 000065AA 51                <1>       push   ecx ; ***
2913 000065AB 89D1              <1>       mov    ecx, edx
2914                            <1> amb_18:
2915 000065AD 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 000065B0 FF0D[98810100]    <1>       dec     dword [free_pages] ; 1 page has been allocated (X = X-1)
2919 000065B6 49                <1>       dec    ecx
2920 000065B7 7404              <1>       jz     short amb_19
2921 000065B9 FEC0              <1>       inc    al
2922 000065BB EBF0              <1>       jmp    short amb_18
2923                            <1> amb_19:
2924 000065BD 59                <1>       pop    ecx ; ***
2925 000065BE 21C9              <1>       and    ecx, ecx ; 0 ?
2926 000065C0 741E              <1>       jz     short amb_22
2927                            <1>       ; 01/04/2016
2928 000065C2 B020              <1>       mov    al, 32
2929                            <1> amb_20:
2930 000065C4 83C304            <1>       add    ebx, 4
2931 000065C7 39C1              <1>       cmp    ecx, eax ; 32
2932 000065C9 7305              <1>       jnb    short amb_21
2933                            <1>       ; ECX < 32
2934 000065CB 28C0              <1>       sub    al, al ; 0
2935 000065CD 50                <1>       push   eax ; 0 ***
2936 000065CE EBDD              <1>       jmp    short amb_18
2937                            <1> amb_21:
2938 000065D0 2905[98810100]    <1>       sub    [free_pages], eax   ; [free_pages] = [free_pages] - 32
2939 000065D6 C70300000000      <1>       mov    dword [ebx], 0         ; reset 32 bits
2940 000065DC 29C1              <1>       sub    ecx, eax ; 32
2941 000065DE 75E4              <1>       jnz    short amb_20
2942                            <1> amb_22:
2943 000065E0 A1[5C8D0100]      <1>       mov    eax, [mem_pg_pos]   ; Beginning address as page number
2944 000065E5 8B0D[548D0100]    <1>       mov    ecx, [mem_aperture] ; Free contiguous page count
2945                            <1>       ; [next_page] update
2946 000065EB 89C2              <1>       mov    edx, eax
2947                            <1>       ; 03/04/2016
2948 000065ED 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 000065F0 80E2FC            <1>       and    dl, 0FCh          ; clear lower 2 bits
2952                            <1>                                ; (to get 32 bit position)
2953 000065F3 3B15[9C810100]    <1>       cmp    edx, [next_page] ; first free page pointer offset
2954 000065F9 7732              <1>       ja     short amb_25
2955 000065FB BB00001000        <1>       mov    ebx, MEM_ALLOC_TBL
2956 00006600 833C1300          <1>       cmp    dword [ebx+edx], 0
2957 00006604 7721              <1>       ja     short amb_24
2958 00006606 89C2              <1>       mov    edx, eax
2959 00006608 01CA              <1>       add    edx, ecx
2960 0000660A C1EA03            <1>       shr    edx, 3
2961 0000660D 80E2FC            <1>       and    dl, 0FCh
2962                            <1> amb_23:
2963 00006610 833C1300          <1>       cmp    dword [ebx+edx], 0
2964 00006614 7711              <1>       ja     short amb_24
2965 00006616 83C204            <1>       add    edx, 4
2966 00006619 3B15[A0810100]    <1>       cmp    edx, [last_page]    ; last page pointer offset
2967 0000661F 76EF              <1>       jna    short amb_23
2968 00006621 8B15[A4810100]    <1>       mov    edx, [first_page]   ; (for) beginning of user's space
2969                            <1> amb_24:
2970 00006627 8915[9C810100]    <1>       mov    [next_page], edx
2971                            <1> amb_25:
2972 0000662D 9C                <1>       pushf
2973 0000662E C1E00C            <1>       shl    eax, PAGE_SHIFT          ; convert to phy. address in bytes
2974 00006631 C1E10C            <1>       shl    ecx, PAGE_SHIFT          ; convert to byte counts
2975 00006634 9D                <1>       popf
2976 00006635 5B                <1>       pop    ebx ; **
2977 00006636 5A                <1>       pop    edx ; *
2978 00006637 C3                <1>       retn
2979                            <1>
2980                            <1> amb_26:     ; set maximum free memory aperture (free memory block size)
2981 00006638 89DA              <1>       mov    edx, ebx ; current address
2982 0000663A 81EA00001000      <1>       sub    edx, MEM_ALLOC_TBL ; MAT beginning address
2983                            <1>       ; 02/04/2016
2984 00006640 C1E203            <1>       shl    edx, 3 ; MAT byte offset * 8 = page number base
2985 00006643 01CA              <1>       add    edx, ecx ; current page number (ecx = 0 to 32)
2986                            <1>       ;
```

```
2987 00006645 A1[548D0100]        <1>       mov    eax, [mem_aperture]
2988 0000664A 21C0                <1>       and    eax, eax
2989 0000664C 7421                <1>         jz      short amb_27
2990 0000664E C705[548D0100]0000- <1>         mov     dword [mem_aperture], 0
2990 00006656 0000                <1>
2991 00006658 3B05[588D0100]      <1>       cmp    eax, [mem_max_aperture]
2992 0000665E 760F                <1>       jna    short amb_27
2993 00006660 A3[588D0100]        <1>       mov    [mem_max_aperture], eax
2994                              <1>       ; 25/04/2017
2995 00006665 A1[5C8D0100]        <1>       mov    eax, [mem_pg_pos]
2996                              <1>       ; EAX = Beginning page number of the max. aperture
2997 0000666A A3[608D0100]        <1>       mov    [mem_max_pg_pos], eax
2998                              <1> amb_27:
2999 0000666F 8915[5C8D0100]      <1>       mov    [mem_pg_pos], edx ; current page
3000                              <1>
3001 00006675 A1[4C8D0100]        <1>       mov    eax, [mem_ipg_count] ; initial (reset) value of page count
3002 0000667A A3[508D0100]        <1>       mov    [mem_pg_count], eax
3003                              <1>
3004 0000667F 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 00006680 52                  <1>       push   edx ; *
3026 00006681 53                  <1>       push   ebx ; **
3027                              <1>
3028 00006682 C1E80C              <1>       shr    eax, PAGE_SHIFT          ; 12
3029 00006685 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 00006688 89C2                <1>       mov    edx, eax
3036 0000668A 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 0000668D 80E2FC              <1>       and    dl, 0FCh          ; clear lower 2 bits
3040                              <1>                                ; (to get 32 bit position)
3041 00006690 3B15[9C810100]      <1>       cmp    edx, [next_page] ; next free page
3042 00006696 7306                <1>       jnb    short damb_1
3043 00006698 8915[9C810100]      <1>       mov    [next_page], edx
3044                              <1> damb_1:
3045 0000669E BB00001000          <1>       mov    ebx, MEM_ALLOC_TBL
3046 000066A3 01D3                <1>       add    ebx, edx
3047 000066A5 83E01F              <1>       and    eax, 1Fh ; 31
3048                              <1>
3049                              <1>       ; 03/04/2016
3050 000066A8 BA20000000          <1>       mov    edx, 32
3051 000066AD 28C2                <1>       sub    dl, al
3052 000066AF 39CA                <1>       cmp    edx, ecx
3053 000066B1 7602                <1>       jna    short damb_2
3054 000066B3 89CA                <1>       mov    edx, ecx
3055                              <1> damb_2:
3056 000066B5 29D1                <1>       sub    ecx, edx
3057 000066B7 51                  <1>       push   ecx ; ***
3058 000066B8 89D1                <1>       mov    ecx, edx
3059                              <1> damb_3:
3060 000066BA 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 000066BD FF05[98810100]      <1>       inc     dword [free_pages]   ; 1 page has been deallocated (X = X+1)
3064 000066C3 49                  <1>       dec    ecx
3065 000066C4 7404                <1>       jz     short damb_4
3066 000066C6 FEC0                <1>       inc    al
3067 000066C8 EBF0                <1>       jmp    short damb_3
3068                              <1> damb_4:
3069 000066CA 59                  <1>       pop    ecx ; ***
3070 000066CB 21C9                <1>       and    ecx, ecx ; 0 ?
3071 000066CD 741E                <1>       jz     short damb_7
3072                              <1>       ; 03/04/2016
3073 000066CF B020                <1>       mov    al, 32
3074                              <1> damb_5:
3075 000066D1 83C304              <1>       add    ebx, 4
3076 000066D4 39C1                <1>       cmp    ecx, eax ; 32
3077 000066D6 7305                <1>       jnb    short damb_6
3078                              <1>       ; ECX < 32
3079 000066D8 28C0                <1>       sub    al, al ; 0
3080 000066DA 50                  <1>       push   eax ; 0 ***
3081 000066DB EBDD                <1>       jmp    short damb_3
3082                              <1> damb_6:
3083 000066DD 0105[98810100]      <1>       add    [free_pages], eax ; [free_pages] = [free_pages] + 32
3084 000066E3 C703FFFFFFFF        <1>       mov    dword [ebx], 0FFFFFFFFh ; set 32 bits
3085 000066E9 29C1                <1>       sub    ecx, eax ; 32
3086 000066EB 75E4                <1>       jnz    short damb_5
3087                              <1> damb_7:
3088 000066ED 5B                  <1>       pop    ebx ; **
3089 000066EE 5A                  <1>       pop    edx ; *
3090 000066EF 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
000066F0 662500F0               <1>       and    ax, PTE_A_CLEAR ; clear page offset
000066F4 50                     <1>       push   eax
                                <1>       ;and   ecx, ecx ; page count
                                <1>       ;jz    dmem_acc_7  ; 'insufficient memory' error
000066F5 89C5                   <1>       mov    ebp, eax
000066F7 81C300004000           <1>       add    ebx, CORE ; 12/05/2017
                                <1> dmem_acc_0:
000066FD 891D[4C980100]         <1>       mov    [base_addr], ebx ; 12/05/2017
00006703 A1[B8030300]           <1>       mov    eax, [u.pgdir] ; page dir address (physical)
00006708 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 ?
0000670D 7324                   <1>       jnc    short dmem_acc_1
                                <1>       ;
0000670F E8B5F4FFFF             <1>       call   allocate_page
00006714 0F82AB000000           <1>         jc     dmem_acc_7  ; 'insufficient memory' error
                                <1>       ;
0000671A E824F5FFFF             <1>       call   clear_page
                                <1>       ; EAX = Physical (base) address of the allocated (new) page
0000671F 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)
00006721 8902                   <1>       mov    [edx], eax ; Let's put the new page directory entry here !
00006723 A1[B8030300]           <1>       mov    eax, [u.pgdir]
00006728 E8B7F5FFFF             <1>       call   get_pte
0000672D 0F8292000000           <1>         jc     dmem_acc_7 ; 'insufficient memory' error
                                <1> dmem_acc_1:
                                <1>       ; EAX = PTE value, EDX = PTE address
00006733 A801                   <1>       test   al, PTE_A_PRESENT
00006735 750D                   <1>       jnz    short dmem_acc_2
00006737 09C0                   <1>       or     eax, eax
00006739 7468                   <1>       jz     short dmem_acc_6   ; Change PTE
0000673B D1E8                   <1>       shr    eax, 1       ; swap disk block (8 sectors) address
                                <1>       ; unlink swap disk block
0000673D E80CFBFFFF             <1>       call   unlink_swap_block
00006742 EB5F                   <1>       jmp    short dmem_acc_6
                                <1>
                                <1> dmem_acc_2:
00006744 A802                   <1>       test   al, PTE_A_WRITE   ; bit 1, writable (r/w) flag
                                <1>                                ; (must be 1)
00006746 7550                   <1>       jnz    short dmem_acc_4
                                <1>       ; Read only -duplicated- page (belongs to a parent or a child)
00006748 66A90002               <1>         test    ax, PTE_DUPLICATED ; Was this page duplicated
                                <1>                                    ; as child's page ?
0000674C 7455                   <1>       jz     short dmem_acc_5 ; Change PTE but don't deallocate the page!
                                <1>
                                <1>       ;push  edi
                                <1>       ;push  esi
                                <1>
0000674E 51                     <1>       push   ecx
                                <1>       ;push  ebx
0000674F 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..
```

```
3196 00006755 89EF              <1>       mov   edi, ebp
3197 00006757 C1EF16            <1>       shr   edi, PAGE_D_SHIFT ; 22
3198                            <1>       ; EDI = page directory entry index (0-1023)
3199 0000675A 89EE              <1>       mov   esi, ebp
3200 0000675C C1EE0C            <1>       shr   esi, PAGE_SHIFT ; 12
3201 0000675F 81E6FF030000      <1>       and   esi, PTE_MASK
3202                            <1>       ; ESI = page table entry index (0-1023)
3203                            <1>
3204 00006765 66C1E702          <1>       shl   di, 2 ; * 4
3205 00006769 01FB              <1>       add   ebx, edi ; PDE offset (for the parent)
3206 0000676B 8B0F              <1>       mov   ecx, [edi]
3207 0000676D F6C101            <1>       test  cl, PDE_A_PRESENT ; present (valid) or not ?
3208 00006770 7425              <1>       jz    short dmem_acc_3    ; parent process does not use this page
3209 00006772 6681E100F0        <1>       and   cx, PDE_A_CLEAR ; 0F000h ; Clear attribute bits
3210 00006777 66C1E602          <1>       shl   si, 2 ; *4
3211 0000677B 01CE              <1>       add   esi, ecx ; PTE offset (for the parent)
3212 0000677D 8B1E              <1>       mov   ebx, [esi]
3213 0000677F F6C301            <1>       test  bl, PTE_A_PRESENT ; present or not ?
3214 00006782 7413              <1>       jz    short dmem_acc_3    ; parent process does not use this page
3215 00006784 662500F0          <1>       and   ax, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
3216 00006788 6681E300F0        <1>       and   bx, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
3217 0000678D 39D8              <1>       cmp   eax, ebx     ; parent's and child's pages are same ?
3218 0000678F 7506              <1>       jne   short dmem_acc_3    ; not same page
3219                            <1>                             ; deallocate the child's page
3220 00006791 800E02            <1>         or     byte [esi], PTE_A_WRITE ; convert to writable page (parent)
3221                            <1>       ;pop  ebx
3222 00006794 59                <1>       pop   ecx
3223 00006795 EB0C              <1>       jmp   short dmem_acc_5
3224                            <1> dmem_acc_3:
3225                            <1>       ;pop  ebx
3226 00006797 59                <1>       pop   ecx
3227                            <1> dmem_acc_4:
3228 00006798 66A90004          <1>       test  ax, PTE_SHARED ; shared or direct memory access indicator
3229 0000679C 7505              <1>       jnz   short dmem_acc_5   ; AVL bit 1 = 1, do not deallocate this page!
3230                            <1>       ;
3231                            <1>       ;and  ax, PTE_A_CLEAR ; 0F000h ; clear lower 12 (attribute) bits
3232 0000679E E804F6FFFF        <1>       call  deallocate_page
3233                            <1> dmem_acc_5:
3234                            <1>       ;pop  esi
3235                            <1>       ;pop  edi
3236                            <1> dmem_acc_6:
3237 000067A3 89E8              <1>       mov   eax, ebp ; physical page (offset=0) address
3238                            <1>       ; EAX = memory page address
3239                            <1>       ; EDX = PTE entry address (physical)
3240 000067A5 660D0704          <1>       or    ax, PTE_A_PRESENT+PTE_A_USER+PTE_A_WRITE+PTE_SHARED
3241                            <1>                   ; present flag, bit 0 = 1
3242                            <1>                   ; user flag, bit 2 = 1
3243                            <1>                   ; writable flag, bit 1 = 1
3244                            <1>                   ; direct memory access flag, bit 10 = 1
3245                            <1>                   ; (This page must not be deallocated!)
3246 000067A9 8902              <1>       mov   [edx], eax  ; Update PTE value
3247 000067AB 49                <1>       dec   ecx ; remain count of contiguous pages
3248 000067AC 741E              <1>       jz    short dmem_acc_8
3249 000067AE 81C500100000      <1>       add   ebp, PAGE_SIZE ; next physical page address
3250                            <1>       ; 22/07/2017
3251                            <1>       ;mov  eax, ebp
3252                            <1>       ; 12/05/2017
3253 000067B4 8B1D[4C980100]    <1>       mov   ebx, [base_addr] ; linear address (virtual+CORE)
3254 000067BA 81C300100000      <1>       add   ebx, PAGE_SIZE     ; next linear address
3255 000067C0 E938FFFFFF        <1>         jmp    dmem_acc_0
3256                            <1> dmem_acc_7:  ; ERROR !
3257 000067C5 C7042404000000    <1>       mov   dword [esp], ERR_MINOR_IM
3258                            <1>             ; Insufficient memory (minor) error!
3259                            <1>             ; Major error = 0 (No protection fault)
3260                            <1>       ; cf = 1
3261                            <1> dmem_acc_8:
3262 000067CC 58                <1>       pop   eax
3263                            <1>       ;pop  edx
3264                            <1>       ;pop  ecx
3265                            <1>       ;pop  ebx
3266                            <1>       ;pop  ebp
3267 000067CD C3                <1>       retn
3268                            <1>
3269                            <1> deallocate_user_pages:
3270                            <1>       ; 20/05/2017
3271                            <1>       ; 15/05/2017
3272                            <1>       ; 20/02/2017
3273                            <1>       ; 19/02/2017 (TRDOS 386 = TRDOS v2.0)
3274                            <1>       ;
3275                            <1>       ; Deallocate virtually contiguous user pages (memory block)
3276                            <1>       ; (caller: 'sysdalloc' system call)
3277                            <1>       ;
3278                            <1>       ; INPUT ->
3279                            <1>       ;     EBX = VIRTUAL ADDRESS (beginning address)
3280                            <1>       ;     ECX = byte count
3281                            <1>       ;     [u.pgdir] = user's page directory
3282                            <1>       ;     [u.ppdir] = parent's page directory
3283                            <1>       ;
3284                            <1>       ; OUTPUT ->
3285                            <1>       ;    If CF = 0
3286                            <1>       ;     EAX = Deallocated memory bytes
3287                            <1>       ;       (Even if shared or read only pages will not be
3288                            <1>       ;        deallocated on M.A.T., this byte count will be
3289                            <1>       ;        returned as virtually deallocated bytes; in fact
3290                            <1>       ;        virtually deallocated user pages * 4096.)
3291                            <1>       ;     EBX = Virtual address (as rounded up)
3292                            <1>       ;    If CF = 1
3293                            <1>       ;     EAX = 0 (there is not any deallocated pages)
3294                            <1>       ;
3295                            <1>       ; Note: Empty page tables will not be deallocated!!!
3296                            <1>       ;     (they will be deallocated at process termination stage)
3297                            <1>       ;
3298                            <1>       ; Modified Registers -> EAX, EDX, ESI, EDI, EBX, ECX, EBP
3299                            <1>       ;
3300 000067CE 89DE              <1>       mov   esi, ebx
```

```
3301 000067D0 89F7              <1>       mov   edi, esi
3302 000067D2 01CF              <1>       add   edi, ecx
3303 000067D4 81C6FF0F0000      <1>       add   esi, PAGE_SIZE - 1  ; 4095 (round up)
3304 000067DA C1EE0C            <1>       shr   esi, PAGE_SHIFT
3305 000067DD C1EF0C            <1>       shr   edi, PAGE_SHIFT
3306 000067E0 89F8              <1>       mov   eax, edi ; end page
3307 000067E2 29F0              <1>       sub   eax, esi ; end page - start page
3308 000067E4 0F86D5000000      <1>       jna   da_u_pd_err  ; < 1
3309 000067EA 89F3              <1>       mov   ebx, esi
3310 000067EC C1E30C            <1>       shl   ebx, PAGE_SHIFT ; virtual address (as rounded up)
3311 000067EF 53                <1>       push  ebx ; *
3312 000067F0 89C1              <1>       mov   ecx, eax ; page count
3313 000067F2 C1E00C            <1>       shl   eax, PAGE_SHIFT ; byte count as adjusted
3314 000067F5 50                <1>       push  eax ; **
3315 000067F6 8B1D[B8030300]    <1>       mov   ebx, [u.pgdir] ; physical addr of user's page dir
3316 000067FC 81C600040000      <1>       add   esi, CORE/PAGE_SIZE
3317 00006802 89F7              <1>       mov   edi, esi
3318 00006804 81E7FF030000      <1>       and   edi, PTE_MASK ; PTE entry in the page table
3319 0000680A 57                <1>       push  edi ; *** ; PTE index (of page directory)
3320 0000680B C1EE0A            <1>       shr   esi, PAGE_D_SHIFT - PAGE_SHIFT ; 22-12=10
3321 0000680E 89F2              <1>       mov   edx, esi
3322                            <1>       ; EDX = PDE index
3323 00006810 C1E602            <1>       shl   esi, 2 ; convert PDE index to dword offset
3324 00006813 01DE              <1>       add   esi, ebx ; add page directory address
3325                            <1> da_u_pd_1:
3326 00006815 AD                <1>       lodsd
3327                            <1>       ;
3328 00006816 89F5              <1>       mov   ebp, esi ; 20/02/2017
3329                            <1>       ; EBP = next PDE address
3330                            <1>       ;
3331 00006818 A801              <1>       test  al, PDE_A_PRESENT ; bit 0, present flag (must be 1)
3332 0000681A 0F8494000000      <1>       jz    da_u_pd_3 ; 20/05/2017
3333 00006820 662500F0          <1>       and   ax, PDE_A_CLEAR ; 0F000h ; clear lower 12 (attribute) bits
3334                            <1>       ; EAX = PHYSICAL (flat) ADDRESS OF THE PAGE TABLE
3335 00006824 8B3C24            <1>       mov   edi, [esp] ; ***
3336                            <1>       ; EDI = PTE index (of complete page directory)
3337                            <1>       ;and  edi, PTE_MASK ; PTE entry in the page table
3338 00006827 C1E702            <1>       shl   edi, 2 ; convert PTE index to dword offset
3339 0000682A 89FE              <1>       mov   esi, edi ; PTE offset in page table (0-4092)
3340 0000682C 01C6              <1>       add   esi, eax ; now, esi points to requested PTE
3341                            <1> da_u_pt_0:
3342 0000682E AD                <1>       lodsd
3343 0000682F A801              <1>       test  al, PTE_A_PRESENT ; bit 0, present flag (must be 1)
3344 00006831 743F              <1>       jz    short da_u_pt_1
3345                            <1>       ;
3346 00006833 A802              <1>       test  al, PTE_A_WRITE   ; bit 1, writable (r/w) flag
3347                            <1>                               ; (must be 1)
3348 00006835 7549              <1>       jnz   short da_u_pt_3
3349                            <1>       ; Read only -duplicated- page (belongs to a parent or a child)
3350 00006837 66A90002          <1>         test    ax, PTE_DUPLICATED ; Was this page duplicated
3351                            <1>                                    ; as child's page ?
3352 0000683B 744E              <1>       jz    short da_u_pt_4 ; Clear PTE but don't deallocate the page!
3353                            <1>       ;
3354                            <1>       ; check the parent's PTE value is read only & same page or not..
3355                            <1>       ; EDX = page directory entry index (0-1023)
3356 0000683D 52                <1>       push  edx ; ****
3357                            <1>       ; EDI = page table entry offset (0-4092)
3358 0000683E 8B1D[BC030300]    <1>       mov   ebx, [u.ppgdir] ; page directory of the parent process
3359 00006844 66C1E202          <1>       shl   dx, 2 ; *4
3360 00006848 01D3              <1>       add   ebx, edx ; PDE address (for the parent)
3361 0000684A 8B13              <1>       mov   edx, [ebx] ; page table address
3362 0000684C F6C201            <1>       test  dl, PDE_A_PRESENT ; present (valid) or not ?
3363 0000684F 742E              <1>       jz    short da_u_pt_2     ; parent process does not use this page
3364 00006851 6681E200F0        <1>       and   dx, PDE_A_CLEAR ; 0F000h ; Clear attribute bits
3365                            <1>       ; EDI = page table entry offset (0-4092)
3366 00006856 01D7              <1>       add   edi, edx     ; PTE address (for the parent)
3367 00006858 8B1F              <1>       mov   ebx, [edi]
3368 0000685A F6C301            <1>       test  bl, PTE_A_PRESENT ; present or not ?
3369 0000685D 7420              <1>       jz    short da_u_pt_2     ; parent process does not use this page
3370 0000685F 662500F0          <1>       and   ax, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
3371 00006863 6681E300F0        <1>       and   bx, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
3372 00006868 39D8              <1>       cmp   eax, ebx     ; parent's and child's pages are same ?
3373 0000686A 7513              <1>       jne   short da_u_pt_2     ; not same page
3374                            <1>                                 ; deallocate the child's page
3375 0000686C 800F02            <1>         or      byte [edi], PTE_A_WRITE ; convert to writable page (parent)
3376 0000686F 5A                <1>       pop   edx ; ****
3377 00006870 EB19              <1>       jmp   short da_u_pt_4
3378                            <1> da_u_pt_1:
3379 00006872 09C0              <1>       or    eax, eax     ; swapped page ?
3380 00006874 741C              <1>       jz    short da_u_pt_5     ; no
3381                            <1>                                 ; yes
3382 00006876 D1E8              <1>       shr   eax, 1
3383 00006878 E8D1F9FFFF        <1>       call  unlink_swap_block ; Deallocate swapped page block
3384                            <1>                                 ; on the swap disk (or in file)
3385 0000687D EB13              <1>       jmp   short da_u_pt_5
3386                            <1> da_u_pt_2:
3387 0000687F 5A                <1>       pop   edx ; ****
3388                            <1> da_u_pt_3:
3389 00006880 66A90004          <1>       test  ax, PTE_SHARED      ; shared or direct memory access indicator
3390 00006884 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 00006886 E81CF5FFFF        <1>       call  deallocate_page ; set the mem allocation bit of this page
3394                            <1> da_u_pt_4:
3395 0000688B 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 00006892 58                <1>       pop   eax ; *** PTE index (of page directory)
3399 00006893 49                <1>       dec   ecx ; remain page count
3400 00006894 7426              <1>       jz    short da_u_pd_4
3401 00006896 40                <1>       inc   eax ; next PTE
3402 00006897 6625FF03          <1>       and   ax, PTE_MASK ; PTE entry index in the page table
3403 0000689B 50                <1>       push  eax ; *** (save again)
3404                            <1>       ;mov  edi, eax
3405                            <1>       ;and  di, PTE_MASK
```

```
                                <1>       ;cmp   edi, PAGE_SIZE / 4 ; 1024
                                <1>       ;jnb   short da_u_pd_2
000068A0 89C7                   <1>       mov    edi, eax
000068A2 C1E702                 <1>       shl    edi, 2 ; convert index to dword offset
                                <1>       ;test  ax, PTE_MASK ; 3FFh
000068A5 09C0                   <1>       or     eax, eax
000068A7 7589                   <1>       jnz    short da_u_pt_0 ; 1-1023
                                <1> da_u_pd_2:
000068A9 42                     <1>       inc    edx
                                <1>       ; 20/05/2017
000068AA 6681E2FF03             <1>       and    dx, PTE_MASK  ; 3FFh
000068AF 740F                   <1>       jz     short da_u_pd_4  ; 0 (1024)
                                <1>       ;cmp   edx, 1024
                                <1>       ;jnb   short da_u_pd_4
000068B1 89EE                   <1>       mov    esi, ebp ; 20/02/2017
000068B3 E961FFFFFF             <1>       jmp    da_u_pd_1
                                <1> da_u_pd_3:
                                <1>       ; 15/05/2017 (empty page directory entry)
000068B8 81E900040000           <1>       sub    ecx, 1024
000068BE 77E9                   <1>       ja     short da_u_pd_2 ; 20/05/2017
                                <1> da_u_pd_4:
000068C0 58                     <1>       pop    eax ; **
000068C1 5B                     <1>       pop    ebx ; *
000068C2 C3                     <1>       retn
                                <1>
                                <1> da_u_pd_err:
000068C3 31C0                   <1>       xor    eax, eax
000068C5 F9                     <1>       stc
000068C6 C3                     <1>       retn
                                <1>
                                <1> allocate_user_pages:
                                <1>       ; 20/05/2017
                                <1>       ; 01/05/2017, 02/05/2017, 15/05/2017
                                <1>       ; 04/03/2017
                                <1>       ; 20/02/2017 (TRDOS 386 = TRDOS v2.0)
                                <1>       ;
                                <1>       ; Allocate physically contiguous user pages (memory block)
                                <1>       ; (caller: 'sysalloc' system call)
                                <1>       ;
                                <1>       ; Note: This procedure does not alloc a page's itself
                                <1>       ;     (page bit) on Memory Allocation Table.
                                <1>       ;     (allocate_memory_block is needed before this proc)
                                <1>       ;
                                <1>       ; INPUT ->
                                <1>       ;     EAX = PHYSICAL ADDRESS (beginning address)
                                <1>       ;     EBX = VIRTUAL ADDRESS (beginning address)
                                <1>       ;     ECX = byte count (>=4096)
                                <1>       ;     [u.pgdir] = user's page directory
                                <1>       ;
                                <1>       ;     Note: All addresses are (must be) already adjusted
                                <1>       ;     to page      borders, otherwise, lower 12bits of addresses
                                <1>       ;     and byte count would be truncated.
                                <1>       ;
                                <1>       ; OUTPUT ->
                                <1>       ;     none
                                <1>       ;
                                <1>       ;     CF = 1 -> insufficient memory error
                                <1>       ;
                                <1>       ; Note: All pages will be allocated in physical page order
                                <1>       ;     from the beginning page address.
                                <1>       ;     * A new page table will be added to the page dir
                                <1>       ;       when the requested PDE is invalid.
                                <1>       ;     * Those pages will not be added to swap queue
                                <1>       ;       because main purpose of this allocation is to
                                <1>       ;       set a direct memory access (DMA controller) buffer.
                                <1>       ;      (Swapping out a page in a DMA buffer would be wrong!)
                                <1>       ;     * Previous content of page tables (PTEs) would be
                                <1>       ;       (should be) deallocated before entering this
                                <1>       ;       procedure. So, new page table entries (PTEs)
                                <1>       ;       directly will be written without checking
                                <1>       ;       their previous content.
                                <1>       ;     * Only solution to increase free memory by removing
                                <1>       ;       that non-swappable memory block is to terminate
                                <1>       ;       the process or to wait until the process will
                                <1>       ;       deallocate that memory block as itself. ('sysdalloc')
                                <1>       ;       (No problem, if the process does not grab all of
                                <1>       ;       -very big amount of- free memory by using
                                <1>       ;       'sysalloc' system call!?)
                                <1>       ;       (Even if the process has grabbed all of free memory,
                                <1>       ;       no problem if the process is not running in
                                <1>       ;       multitasking mode. No problem in multitasking
                                <1>       ;       mode if there is not another process which is running
                                <1>       ;       or waiting or sleeping for an event as it's pages
                                <1>       ;       are swapped-out. But a new process can not start to
                                <1>       ;       run if all of free memory has beeen allocated
                                <1>       ;       by running processes. Deallocation -'sysdalloc'-
                                <1>       ;       or terminate a running process is needed
                                <1>       ;       in order to run a new process.)
                                <1>       ;
                                <1>       ; Modified Registers -> EAX, EDX, ESI, EDI, EBX, ECX, EBP
                                <1>       ;
                                <1>
                                <1>       ; 01/05/2017
000068C7 662500F0               <1>       and    ax, ~PAGE_OFF
000068CB 6681E300F0             <1>       and    bx, ~PAGE_OFF
                                <1>       ; 02/05/2017
000068D0 BD00F0FFFF             <1>       mov    ebp, 0FFFFF000h ; 4 Giga Bytes - 4096 Bytes (for Stack)
000068D5 C1E90C                 <1>       shr    ecx, PAGE_SHIFT ; page count
000068D8 83F901                 <1>       cmp    ecx, 1
000068DB 7251                   <1>       jb     short a_u_im_retn
000068DD 89C2                   <1>       mov    edx, eax
000068DF 01CA                   <1>       add    edx, ecx
000068E1 724B                   <1>       jc     short a_u_im_retn
000068E3 39D5                   <1>       cmp    ebp, edx
000068E5 7247                   <1>       jb     short a_u_im_retn
```

```
3511 000068E3 89DA                <1>       mov    edx, ebx
3512 000068E5 81C200004000        <1>       add    edx, CORE
3513 000068EB 723D                <1>       jc     short a_u_im_retn
3514 000068ED 01CA                <1>       add    edx, ecx
3515 000068EF 7239                <1>       jc     short a_u_im_retn
3516 000068F1 39D5                <1>       cmp    ebp, edx
3517 000068F3 7235                <1>       jb     short a_u_im_retn
3518                              <1>       ;
3519 000068F5 89C5                <1>       mov    ebp, eax ; physical address
3520 000068F7 89DE                <1>       mov    esi, ebx
3521 000068F9 81C600004000        <1>       add    esi, CORE ; start of user's memory (4M)
3522 000068FF C1EE0C              <1>       shr    esi, PAGE_SHIFT ; higher 20 bits of the linear address
3523                              <1>       ;shr   ecx, PAGE_SHIFT ; page count
3524 00006902 8B1D[B8030300]      <1>       mov    ebx, [u.pgdir] ; physical addr of user's page dir
3525 00006908 89F7                <1>       mov    edi, esi
3526 0000690A 81E7FF030000        <1>       and    edi, PTE_MASK ; PTE entry index in the page table
3527 00006910 57                  <1>       push   edi  ; * ; PTE index (in page directory)
3528 00006911 C1EE0A              <1>       shr    esi, PAGE_D_SHIFT - PAGE_SHIFT ; 22-12=10
3529 00006914 89F2                <1>       mov    edx, esi
3530                              <1>       ; EDX = PDE index
3531 00006916 C1E602              <1>       shl    esi, 2 ; convert PDE index to dword offset
3532 00006919 01DE                <1>       add    esi, ebx ; add page directory address
3533                              <1> a_u_pd_0:
3534 0000691B AD                  <1>       lodsd
3535                              <1>       ;
3536 0000691C 89F3                <1>       mov    ebx, esi ; next PDE address
3537                              <1>       ;
3538 0000691E A801                <1>       test   al, PDE_A_PRESENT ; bit 0, present flag (must be 1)
3539 00006920 7513                <1>       jnz    short a_u_pd_2
3540                              <1>       ;
3541                              <1>       ; empty PDE (it does not point to valid page table address)
3542 00006922 E8A2F2FFFF          <1>       call   allocate_page  ; (allocate a new page table)
3543 00006927 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 00006929 5E                  <1>       pop    esi ; *
3547                              <1> a_u_im_retn:
3548 0000692A 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 0000692B E813F3FFFF          <1>       call   clear_page ; Clear page content
3553                              <1>       ;
3554 00006930 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 00006932 8946FC              <1>       mov    [esi-4], eax
3558                              <1> a_u_pd_2:
3559 00006935 662500F0            <1>       and    ax, PDE_A_CLEAR ; 0F000h ; clear lower 12 (attribute) bits
3560                              <1>       ; EAX = PHYSICAL (flat) ADDRESS OF THE PAGE TABLE
3561 00006939 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 0000693C 89FE                <1>       mov    esi, edi ; PTE index in page table (0-1023)
3566 0000693E C1E702              <1>       shl    edi, 2 ; convert PTE index to dword offset
3567 00006941 01C7                <1>       add    edi, eax ; now, edi points to requested PTE
3568                              <1> a_u_pt_0:
3569                              <1>       ; 02/05/2017
3570 00006943 8B07                <1>       mov    eax, [edi]
3571                              <1>       ;
3572 00006945 A801                <1>       test   al, PTE_A_PRESENT ; bit 0, present flag (must be 1)
3573 00006947 7445                <1>       jz     short a_u_pt_1
3574                              <1>       ;
3575 00006949 A802                <1>       test   al, PTE_A_WRITE   ; bit 1, writable (r/w) flag
3576                              <1>                                ; (must be 1)
3577 0000694B 7550                <1>       jnz    short a_u_pt_3
3578                              <1>       ; Read only -duplicated- page (belongs to a parent or a child)
3579 0000694D 66A90002            <1>         test    ax, PTE_DUPLICATED ; Was this page duplicated
3580                              <1>                                    ; as child's page ?
3581 00006951 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 00006953 52                  <1>       push   edx ; **
3586 00006954 53                  <1>       push   ebx ; ***
3587                              <1>       ; ESI = page table entry index (0-1023)
3588                              <1>       ;push  esi ; **** ; 20/05/2017
3589 00006955 8B1D[BC030300]      <1>       mov    ebx, [u.ppgdir] ; page directory of the parent process
3590 0000695B 66C1E202            <1>       shl    dx, 2 ; *4
3591 0000695F 01D3                <1>       add    ebx, edx ; PTE address,0 (for the parent)
3592 00006961 8B13                <1>       mov    edx, [ebx] ; page table address
3593 00006963 F6C201              <1>       test   dl, PDE_A_PRESENT ; present (valid) or not ?
3594 00006966 7433                <1>       jz     short a_u_pt_2      ; parent process does not use this page
3595 00006968 6681E200F0          <1>       and    dx, PDE_A_CLEAR ; 0F000h ; Clear attribute bits
3596 0000696D 66C1E602            <1>       shl    si, 2 ; *4
3597                              <1>       ; ESI = page table entry offset (0-4092)
3598 00006971 01D6                <1>       add    esi, edx     ; PTE address (for the parent)
3599 00006973 8B1E                <1>       mov    ebx, [esi]
3600 00006975 F6C301              <1>       test   bl, PTE_A_PRESENT ; present or not ?
3601 00006978 7421                <1>       jz     short a_u_pt_2      ; parent process does not use this page
3602 0000697A 662500F0            <1>       and    ax, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
3603 0000697E 6681E300F0          <1>       and    bx, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
3604 00006983 39D8                <1>       cmp    eax, ebx     ; parent's and child's pages are same ?
3605 00006985 7514                <1>       jne    short a_u_pt_2      ; not same page
3606                              <1>                                ; deallocate the child's page
3607 00006987 800E02              <1>         or     byte [esi], PTE_A_WRITE ; convert to writable page (parent)
3608                              <1>       ;pop   esi ; **** ; 20/05/2017
3609 0000698A 5B                  <1>       pop    ebx ; ***
3610 0000698B 5A                  <1>       pop    edx ; **
3611 0000698C EB1A                <1>       jmp    short a_u_pt_4
3612                              <1> a_u_pt_1:
3613 0000698E 09C0                <1>       or     eax, eax     ; swapped page ?
3614 00006990 7416                <1>       jz     short a_u_pt_4      ; no
3615                              <1>                                ; yes
```

```
3616 00006992 D1E8              <1>       shr    eax, 1
3617 00006994 E8B5F8FFFF        <1>       call   unlink_swap_block ; Deallocate swapped page block
3618                            <1>                                ; on the swap disk (or in file)
3619 00006999 EB0D              <1>       jmp    short a_u_pt_4
3620                            <1> a_u_pt_2:
3621                            <1>       ;pop   esi ; **** ; 20/05/2017
3622 0000699B 5B                <1>       pop    ebx ; ***
3623 0000699C 5A                <1>       pop    edx ; **
3624                            <1> a_u_pt_3:
3625 0000699D 66A90004          <1>       test   ax, PTE_SHARED      ; shared or direct memory access indicator
3626 000069A1 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 000069A3 E8FFF3FFFF        <1>       call   deallocate_page ; set the mem allocation bit of this page
3630                            <1>       ;
3631                            <1> a_u_pt_4:
3632 000069A8 89E8              <1>       mov    eax, ebp ; physical address
3633 000069AA 0C07              <1>       or     al, PTE_A_PRESENT + PTE_A_WRITE + PTE_A_USER ; 04/03/2017
3634 000069AC AB                <1>       stosd
3635 000069AD 5E                <1>       pop    esi ; * ; 20/05/2017
3636 000069AE 49                <1>       dec    ecx ; remain page count
3637 000069AF 7417              <1>       jz     short a_u_pd_5
3638 000069B1 81C500100000      <1>       add    ebp, PAGE_SIZE
3639 000069B7 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 000069B8 6681E6FF03        <1>       and    si, PTE_MASK ; 3FFh (0 to 1023)
3644 000069BD 56                <1>       push   esi ; *
3645 000069BE 7583              <1>       jnz    short a_u_pt_0 ; > 0 (<1024)
3646                            <1> a_u_pd_3:
3647 000069C0 42                <1>       inc    edx
3648                            <1> ;     cmp    edx, 1024
3649                            <1> ;     jnb    short a_u_pd_4 ; 02/05/2017 (error!, ecx > 0)
3650 000069C1 89DE              <1>       mov    esi, ebx ; the next PDE address
3651 000069C3 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 000069C8 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 000069C9 8B3D[2C120300]    <1>       mov    edi, [LFB_ADDR]
3676 000069CF 8B15[30120300]    <1>       mov    edx, [LFB_SIZE]
3677                            <1>
3678 000069D5 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 000069D8 66C1E702          <1>       shl    di, 2  ; * 4 for offset
3683                            <1>
3684                            <1>       ;add   edx, 4095
3685 000069DC C1EA0C            <1>       shr    edx, 12      ; convert LFB size to LFB page count
3686                            <1>
3687 000069DF 89D1              <1>       mov    ecx, edx ; * ; LFB page count
3688                            <1>
3689 000069E1 81C1FF030000      <1>       add    ecx, 1023 ; page count + 1023
3690 000069E7 C1E90A            <1>       shr    ecx, 10 ; convert to page directory entry count
3691                            <1>                      ; (page table count)
3692 000069EA 51                <1>       push   ecx ; **
3693 000069EB C1E10C            <1>       shl    ecx, 12 ; convert to byte count
3694                            <1>
3695 000069EE 31C0              <1>       xor    eax, eax ; first available pages
3696                            <1>
3697                            <1>       ; allocate contiguous memory block for these kernel pages
3698                            <1>
3699 000069F0 E87EFAFFFF        <1>       call   allocate_memory_block
3700                            <1>       ; eax = start address of (contiguous) memory block
3701 000069F5 59                <1>       pop    ecx ; ** ; PDE count
3702 000069F6 7301              <1>       jnc    short a_lfb_k_1
3703                            <1>       ; error (cf=1)
3704 000069F8 C3                <1>       retn
3705                            <1> a_lfb_k_1:
3706                            <1>       ; Allocate (new) page tables in kernel's page directory
3707 000069F9 51                <1>       push   ecx ; PDE (page table) count
3708 000069FA 50                <1>       push   eax ; start address of contiguous memory pages
3709                            <1>                  ; (at page boundary)
3710                            <1>       ; edi = 1st page directory entry offset
3711 000069FB 033D[90810100]    <1>       add    edi, [k_page_dir] ; Kernel's Page Dir Address
3712                            <1> a_lfb_k_2:
3713 00006A01 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 00006A05 AB                <1>       stosd
3717 00006A06 0500100000        <1>       add    eax, 4096
3718 00006A0B E2F4              <1>       loop   a_lfb_k_2
3719                            <1>
3720 00006A0D 5F                <1>       pop    edi ; start addr of contiguous memory pages
```

```
3721 00006A0E 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 00006A0F C1E10A              <1>       shl    ecx, 10 ; * 1024, convert to (total) PTE count
3727                              <1>
3728 00006A12 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 00006A17 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 00006A1B AB                  <1>       stosd
3736 00006A1C 4A                  <1>       dec    edx
3737 00006A1D 7408                <1>       jz     short a_lfb_k_4 ; LFB size has been completed (!?)
3738 00006A1F 0500100000          <1>       add    eax, 4096
3739 00006A24 E2F1                <1>       loop   a_lfb_k_3
3740                              <1>
3741 00006A26 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 00006A27 31C0                <1>       xor    eax, eax ; clear page table entry (empty)
3747 00006A29 F3AB                <1>       rep    stosd
3748 00006A2B 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
2888                                  %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> ;
```

```
                                <1> ; Derived from 'IBM PC-AT' BIOS source code (1985)
                                <1> ; ****************************************************************************
                                <1>
                                <1> ; TRDOS 386  (TRDOS v2.0) Kernel - TIMER & REAL TIME CLOCK (BIOS) FUNCTIONS
                                <1>
                                <1> ; IBM PC-AT BIOS Source Code ('BIOS2.ASM')
                                <1> ; TITLE BIOS2 ---- 06/10/85 BIOS INTERRUPT ROUTINES
                                <1>
                                <1> ;
                                <1> ; ///////// TIMER (& REAL TIME CLOCK) FUNCTIONS ///////////////
                                <1>
                                <1> int1Ah:
                                <1>       ; 29/01/2016
                                <1>       ; 17/01/2016 (TRDOS 386 = TRDOS v2.0)
00006A2C 9C                     <1>       pushfd
00006A2D 0E                     <1>       push   cs
00006A2E E801000000             <1>       call   TIME_OF_DAY_1
00006A33 C3                     <1>       retn
                                <1>
                                <1> ;--- INT  1A H -- (TIME OF DAY) ------------------------------------------------
                                <1> ;       THIS BIOS ROUTINE ALLOWS THE CLOCKS TO BE SET OR READ                :
                                <1> ;                                                                            :
                                <1> ; PARAMETERS:                                                                :
                                <1> ;     (AH) = 00H  READ THE CURRENT SETTING AND RETURN WITH,                  :
                                <1> ;                       (CX) = HIGH PORTION OF COUNT                         :
                                <1> ;                       (DX) = LOW PORTION OF COUNT                          :
                                <1> ;                       (AL) = 0 TIMER HAS NOT PASSED 24 HOURS SINCE LAST READ :
                                <1> ;                              1 IF ON ANOTHER DAY. (RESET TO ZERO AFTER READ) :
                                <1> ;                                                                            :
                                <1> ;     (AH) = 01H  SET THE CURRENT CLOCK USING,                               :
                                <1> ;                 (CX) = HIGH PORTION OF COUNT                               :
                                <1> ;                 (DX) = LOW PORTION OF COUNT.                               :
                                <1> ;                                                                            :
                                <1> ;              NOTE: COUNTS OCCUR AT THE RATE OF 1193180/65536 COUNTS/SECOND :
                                <1> ;                           (OR ABOUT 18.2 PER SECOND -- SEE EQUATES)        :
                                <1> ;                                                                            :
                                <1> ;     (AH) = 02H  READ THE REAL TIME CLOCK AND RETURN WITH,                  :
                                <1> ;                       (CH) = HOURS IN BCD (00-23)                          :
                                <1> ;                       (CL) = MINUTES IN BCD (00-59)                        :
                                <1> ;                       (DH) = SECONDS IN BCD (00-59)                        :
                                <1> ;                       (DL) = DAYLIGHT SAVINGS ENABLE (00-01)               :
                                <1> ;                                                                            :
                                <1> ;     (AH) = 03H  SET THE REAL TIME CLOCK USING,                             :
                                <1> ;                     (CH) = HOURS IN BCD (00-23)                            :
                                <1> ;                     (CL) = MINUTES IN BCD (00-59)                          :
                                <1> ;                     (DH) = SECONDS IN BCD (00-59)                          :
                                <1> ;                     (DL) = 01 IF DAYLIGHT SAVINGS ENABLE OPTION, ELSE 00.  :
                                <1> ;                                                                            :
                                <1> ;            NOTE: (DL) = 00 IF DAYLIGHT SAVINGS TIME ENABLE IS NOT ENABLED. :
                                <1> ;                  (DL) = 01 ENABLES TWO SPECIAL UPDATES THE LAST SUNDAY IN  :
                                <1> ;               APRIL   (1:59:59 --> 3:00:00 AM) AND THE LAST SUNDAY IN      :
                                <1> ;                  OCTOBER (1:59:59 --> 1:00:00 AM) THE FIRST TIME.          :
                                <1> ;                                                                            :
                                <1> ;     (AH) = 04H  READ THE DATE FROM THE REAL TIME CLOCK AND RETURN WITH,    :
                                <1> ;                       (CH) = CENTURY IN BCD (19 OR 20)                     :
                                <1> ;                       (CL) = YEAR IN BCD (00-99)                           :
                                <1> ;                       (DH) = MONTH IN BCD (01-12)                          :
                                <1> ;                       (DL) = DAY IN BCD (01-31).                           :
                                <1> ;                                                                            :
                                <1> ;     (AH) = 05H  SET THE DATE INTO THE REAL TIME CLOCK USING,               :
                                <1> ;                     (CH) = CENTURY IN BCD (19 OR 20)                       :
                                <1> ;                     (CL) = YEAR IN BCD (00-99)                             :
                                <1> ;                     (DH) = MONTH IN BCD (01-12)                            :
                                <1> ;                     (DL) = DAY IN BCD (01-31).                             :
                                <1> ;                                                                            :
                                <1> ;     (AH) = 06H  SET THE ALARM TO INTERRUPT AT SPECIFIED TIME,              :
                                <1> ;                     (CH) = HOURS IN BCD (00-23 (OR FFH))                   :
                                <1> ;                     (CL) = MINUTES IN BCD (00-59 (OR FFH))                 :
                                <1> ;                     (DH) = SECONDS IN BCD (00-59 (OR FFH))                 :
                                <1> ;                                                                            :
                                <1> ;     (AH) = 07H  RESET THE ALARM INTERRUPT FUNCTION.                        :
                                <1> ;                                                                            :
                                <1> ; NOTES: FOR ALL RETURNS CY= 0 FOR SUCCESSFUL OPERATION.                     :
                                <1> ;        FOR (AH)= 2, 4, 6 - CARRY FLAG SET IF REAL TIME CLOCK NOT OPERATING. :
                                <1> ;        FOR (AH)= 6 - CARRY FLAG SET IF ALARM ALREADY ENABLED.              :
                                <1> ;        FOR THE ALARM FUNCTION (AH = 6) THE USER MUST SUPPLY A ROUTINE AND  :
                                <1> ;         INTERCEPT THE CORRECT ADDRESS IN THE VECTOR TABLE FOR INTERRUPT 4AH.      :
                                <1> ;         USE 0FFH FOR ANY "DO NOT CARE" POSITION FOR INTERVAL INTERRUPTS.   :
                                <1> ;        INTERRUPTS ARE DISABLED DURING DATA MODIFICATION.                   :
                                <1> ;        AH & AL ARE RETURNED MODIFIED AND NOT DEFINED EXCEPT WHERE INDICATED.      :
                                <1> ;--------------------------------------------------------------------------------
                                <1>
                                <1> ; 15/01/2017
                                <1> ; 14/01/2017
                                <1> ; 07/01/2017
                                <1> ; 02/01/2017
                                <1> ; 29/05/2016
                                <1> ; 29/01/2016
                                <1> ; 17/01/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>
                                <1> ; 29/05/2016
                                <1> ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
                                <1> int35h:  ; Date/Time functions
                                <1>
                                <1> TIME_OF_DAY_1:
                                <1>       ;sti                      ; INTERRUPTS BACK ON
                                <1>       ; 29/05/2016
00006A34 80642408FE             <1>       and    byte [esp+8], 11111110b   ; clear carry bit of eflags register
                                <1>       ;
00006A39 80FC08                 <1>       cmp    ah, (RTC_TBE-RTC_TB)/4    ; CHECK IF COMMAND IN VALID RANGE (0-7)
00006A3C F5                     <1>       cmc                       ; COMPLEMENT CARRY FOR ERROR EXIT
                                <1>       ; (*) jc short TIME_9           ; EXIT WITH CARRY = 1 IF NOT VALID
00006A3D 721A                   <1>       jc     short _TIME_9 ; 29/05/2016
                                <1>
00006A3F 1E                     <1>       push   ds
```

```
00006A40 56                  <1>       push   esi
00006A41 66BE1000            <1>       mov    si, KDATA           ; kernel data segment
00006A45 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>       ;
00006A47 C0E402              <1>       shl    ah, 2               ; convert function to dword offset
00006A4A 0FB6F4              <1>       movzx  esi, ah                   ; PLACE INTO ADDRESSING REGISTER
                             <1>       ;cli                       ; NO INTERRUPTS DURING TIME FUNCTIONS
00006A4D FF96[5F6A0000]      <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
00006A53 B400                <1>       mov    ah, 0               ; CLEAR (AH) TO ZERO
00006A55 5E                  <1>       pop    esi                 ; RECOVER USERS REGISTER
00006A56 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
00006A57 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
00006A59 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:
00006A5E 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:
00006A5F [7F6A0000]          <1>       dd     RTC_00              ; 0 = READ CURRENT CLOCK COUNT
00006A63 [926A0000]          <1>       dd     RTC_10              ; 1 = SET CLOCK COUNT
00006A67 [A06A0000]          <1>       dd     RTC_20              ; 2 = READ THE REAL TIME CLOCK TIME
00006A6B [CF6A0000]          <1>       dd     RTC_30              ; 3 = SET REAL TIME CLOCK TIME
00006A6F [116B0000]          <1>       dd     RTC_40              ; 4 = READ THE REAL TIME CLOCK DATE
00006A73 [3E6B0000]          <1>       dd     RTC_50              ; 5 = SET REAL TIME CLOCK DATE
00006A77 [8B6B0000]          <1>       dd     RTC_60              ; 6 = SET THE REAL TIME CLOCK ALARM
00006A7B [DE6B0000]          <1>       dd     RTC_70              ; 7 = RESET ALARM
                             <1>
                             <1> RTC_TBE      equ    $
                             <1>
                             <1> RTC_00:                          ; READ TIME COUNT
00006A7F A0[14820100]        <1>       mov    al, [TIMER_OFL]           ; GET THE OVERFLOW FLAG
00006A84 C605[14820100]00    <1>       mov    byte [TIMER_OFL], 0 ; AND THEN RESET THE OVERFLOW FLAG
00006A8B 8B0D[10820100]      <1>         mov     ecx, [TIMER_LH]         ; GET COUNT OF TIME
00006A91 C3                  <1>       retn
                             <1>
                             <1> RTC_10:                          ; SET TIME COUNT
00006A92 890D[10820100]      <1>         mov     [TIMER_LH], ecx         ; SET TIME COUNT
00006A98 C605[14820100]00    <1>       mov    byte [TIMER_OFL], 0 ; RESET OVERFLOW FLAG
00006A9F C3                  <1>       retn                        ; RETURN WITH NO CARRY
                             <1>
                             <1> RTC_20:                          ; GET RTC TIME
00006AA0 E8EB010000          <1>       call   UPD_IPR                   ; CHECK FOR UPDATE IN PROCESS
00006AA5 7227                <1>       jc     short RTC_29        ; EXIT IF ERROR (CY= 1)
                             <1>
00006AA7 B000                <1>       mov    al, CMOS_SECONDS    ; SET ADDRESS OF SECONDS
00006AA9 E8FD010000          <1>       call   CMOS_READ           ; GET SECONDS
00006AAE 88C6                <1>       mov    dh, al              ; SAVE
00006AB0 B00B                <1>       mov    al, CMOS_REG_B           ; ADDRESS ALARM REGISTER
00006AB2 E8F4010000          <1>       call   CMOS_READ           ; READ CURRENT VALUE OF DSE BIT
00006AB7 2401                <1>       and    al, 00000001b       ; MASK FOR VALID DSE BIT
00006AB9 88C2                <1>       mov    dl, al              ; SET [DL] TO ZERO FOR NO DSE BIT
00006ABB B002                <1>       mov    al, CMOS_MINUTES    ; SET ADDRESS OF MINUTES
00006ABD E8E9010000          <1>       call   CMOS_READ           ; GET MINUTES
00006AC2 88C1                <1>       mov    cl, al              ; SAVE
00006AC4 B004                <1>         mov     al, CMOS_HOURS          ; SET ADDRESS OF HOURS
00006AC6 E8E0010000          <1>       call   CMOS_READ           ; GET HOURS
00006ACB 88C5                <1>       mov    ch, al              ; SAVE
00006ACD F8                  <1>       clc                        ; SET CY= 0
                             <1> RTC_29:
00006ACE C3                  <1>       retn                        ; RETURN WITH RESULT IN CARRY FLAG
                             <1>
                             <1> RTC_30:                          ; SET RTC TIME
00006ACF E8BC010000          <1>       call   UPD_IPR                   ; CHECK FOR UPDATE IN PROCESS
```

```
00006AD4 7305              <1>       jnc   short RTC_35        ; GO AROUND IF CLOCK OPERATING
00006AD6 E817010000        <1>       call  RTC_STA                   ; ELSE TRY INITIALIZING CLOCK
                           <1> RTC_35:
00006ADB 88F4              <1>       mov   ah, dh              ; GET TIME BYTE - SECONDS
00006ADD B000              <1>       mov   al, CMOS_SECONDS    ; ADDRESS SECONDS
00006ADF E8E0010000        <1>       call  CMOS_WRITE          ; UPDATE SECONDS
00006AE4 88CC              <1>       mov   ah, cl              ; GET TIME BYTE - MINUTES
00006AE6 B002              <1>       mov   al, CMOS_MINUTES    ; ADDRESS MINUTES
00006AE8 E8D7010000        <1>       call  CMOS_WRITE          ; UPDATE MINUTES
00006AED 88EC              <1>       mov   ah, ch              ; GET TIME BYTE - HOURS
00006AEF B004              <1>       mov   al, CMOS_HOURS            ; ADDRESS HOURS
00006AF1 E8CE010000        <1>       call  CMOS_WRITE          ; UPDATE ADDRESS
                           <1>       ;mov  al, CMOS_REG_B            ; ADDRESS ALARM REGISTER
                           <1>       ;mov  ah, al
00006AF6 66B80B0B          <1>       mov   ax, CMOS_REG_B * 257
00006AFA E8AC010000        <1>       call  CMOS_READ           ; READ CURRENT TIME
00006AFF 2462              <1>       and   al, 01100010b       ; MASK FOR VALID BIT POSITIONS
00006B01 0C02              <1>       or    al, 00000010b       ; TURN ON 24 HOUR MODE
00006B03 80E201            <1>       and   dl, 00000001b       ; USE ONLY THE DSE BIT
00006B06 08D0              <1>       or    al, dl              ; GET DAY LIGHT SAVINGS TIME BIT (OSE)
00006B08 86E0              <1>       xchg  ah, al              ; PLACE IN WORK REGISTER AND GET ADDRESS
00006B0A E8B5010000        <1>       call  CMOS_WRITE          ; SET NEW ALARM SITS
00006B0F F8                <1>       clc                       ; SET CY= 0
00006B10 C3                <1>       retn                      ; RETURN WITH CY= 0
                           <1>
                           <1> RTC_40:                         ; GET RTC DATE
00006B11 E87A010000        <1>       call  UPD_IPR                   ; CHECK FOR UPDATE IN PROCESS
00006B16 7225              <1>       jc    short RTC_49        ; EXIT IF ERROR (CY= 1)
                           <1>
00006B18 B007              <1>       mov   al, CMOS_DAY_MONTH  ; ADDRESS DAY OF MONTH
00006B1A E88C010000        <1>       call  CMOS_READ           ; READ DAY OF MONTH
00006B1F 88C2              <1>       mov   dl, al              ; SAVE
00006B21 B008              <1>       mov   al, CMOS_MONTH            ; ADDRESS MONTH
00006B23 E883010000        <1>       call  CMOS_READ           ; READ MONTH
00006B28 88C6              <1>       mov   dh, al              ; SAVE
00006B2A B009              <1>       mov   al, CMOS_YEAR       ; ADDRESS YEAR
00006B2C E87A010000        <1>       call  CMOS_READ           ; READ YEAR
00006B31 88C1              <1>       mov   cl, al              ; SAVE
00006B33 B032              <1>       mov   al, CMOS_CENTURY    ; ADDRESS CENTURY LOCATION
00006B35 E871010000        <1>       call  CMOS_READ           ; GET CENTURY BYTE
00006B3A 88C5              <1>       mov   ch, al              ; SAVE
00006B3C F8                <1>       clc                       ; SET CY=0
                           <1> RTC_49:
00006B3D C3                <1>       retn                      ; RETURN WITH RESULTS IN CARRY FLAG
                           <1>
                           <1> RTC_50:                         ; SET RTC DATE
00006B3E E84D010000        <1>       call  UPD_IPR                   ; CHECK FOR UPDATE IN PROCESS
00006B43 7305              <1>       jnc   short RTC_55        ; GO AROUND IF NO ERROR
00006B45 E8A8000000        <1>       call  RTC_STA                   ; ELSE INITIALIZE CLOCK
                           <1> RTC_55:
00006B4A 66B80600          <1>       mov   ax, CMOS_DAY_WEEK   ; ADDRESS OF DAY OF WEEK BYTE
00006B4E E871010000        <1>       call  CMOS_WRITE          ; LOAD ZEROS TO DAY OF WEEK
00006B53 88D4              <1>       mov   ah, dl              ; GET DAY OF MONTH BYTE
00006B55 B007              <1>       mov   al, CMOS_DAY_MONTH  ; ADDRESS DAY OF MONTH BYTE
00006B57 E868010000        <1>       call  CMOS_WRITE          ; WRITE OF DAY OF MONTH REGISTER
00006B5C 88F4              <1>       mov   ah, dh              ; GET MONTH
00006B5E B008              <1>       mov   al, CMOS_MONTH            ; ADDRESS MONTH BYTE
00006B60 E85F010000        <1>       call  CMOS_WRITE          ; WRITE MONTH REGISTER
00006B65 88CC              <1>       mov   ah, cl              ; GET YEAR BYTE
00006B67 B009              <1>       mov   al, CMOS_YEAR       ; ADDRESS YEAR REGISTER
00006B69 E856010000        <1>       call  CMOS_WRITE          ; WRITE YEAR REGISTER
00006B6E 88EC              <1>       mov   ah, ch              ; GET CENTURY BYTE
00006B70 B032              <1>       mov   al, CMOS_CENTURY    ; ADDRESS CENTURY BYTE
00006B72 E84D010000        <1>       call  CMOS_WRITE          ; WRITE CENTURY LOCATION
                           <1>       ;mov  al, CMOS_REG_B            ; ADDRESS ALARM REGISTER
                           <1>       ;mov  ah, al
00006B77 66B80B0B          <1>       mov   ax, CMOS_REG_B * 257
00006B7B E82B010000        <1>       call  CMOS_READ           ; READ WIRRENT SETTINGS
00006B80 247F              <1>       and   al, 07Fh            ; CLEAR 'SET BIT'
00006B82 86E0              <1>       xchg  ah, al              ; MOVE TO WORK REGISTER
00006B84 E83B010000        <1>       call  CMOS_WRITE          ; AND START CLOCK UPDATING
00006B89 F8                <1>       clc                       ; SET CY= 0
00006B8A C3                <1>       retn                      ; RETURN CY=0
                           <1>
                           <1> RTC_60:                         ; SET RTC ALARM
00006B8B B00B              <1>       mov   al, CMOS_REG_B            ; ADDRESS ALARM
00006B8D E819010000        <1>       call  CMOS_READ           ; READ ALARM REGISTER
00006B92 A820              <1>       test  al, 20h                   ; CHECK FOR ALARM ALREADY ENABLED
00006B94 F9                <1>       stc                       ; SET CARRY IN CASE OF ERROR
00006B95 7542              <1>       jnz   short RTC_69        ; ERROR EXIT IF ALARM SET
00006B97 E8F4000000        <1>       call  UPD_IPR                   ; CHECK FOR UPDATE IN PROCESS
00006B9C 7305              <1>       jnc   short RTC_65        ; SKIP INITIALIZATION IF NO ERROR
00006B9E E84F000000        <1>       call  RTC_STA                   ; ELSE INITIALIZE CLOCK
                           <1> RTC_65:
00006BA3 88F4              <1>       mov   ah, dh              ; GET SECONDS BYTE
00006BA5 B001              <1>       mov   al, CMOS_SEC_ALARM  ; ADDRESS THE SECONDS ALARM REGISTER
00006BA7 E818010000        <1>       call  CMOS_WRITE          ; INSERT SECONDS
00006BAC 88CC              <1>       mov   ah, cl              ; GET MINUTES PARAMETER
00006BAE B003              <1>       mov   al, CMOS_MIN_ALARM  ; ADDRESS MINUTES ALARM REGISTER
00006BB0 E80F010000        <1>       call  CMOS_WRITE          ; INSERT MINUTES
00006BB5 88EC              <1>       mov   ah, ch              ; GET HOURS PARAMETER
00006BB7 B005              <1>       mov   al, CMOS_HR_ALARM   ; ADDRESS HOUR ALARM REGISTER
00006BB9 E806010000        <1>       call  CMOS_WRITE          ; INSERT HOURS
00006BBE E4A1              <1>       in    al, INTB01          ; READ SECOND INTERRUPT MASK REGISTER
00006BC0 24FE              <1>       and   al, 0FEh            ; ENABLE ALARM TIMER BIT (CY= 0)
00006BC2 E6A1              <1>       out   INTB01, al          ; WRITE UPDATED MASK
                           <1>       ;mov  al, CMOS_REG_B            ; ADDRESS ALARM REGISTER
                           <1>       ;mov  ah, al
00006BC4 66B80B0B          <1>       mov   ax, CMOS_REG_B * 257
00006BC8 E8DE000000        <1>       call  CMOS_READ           ; READ CURRENT ALARM REGISTER
00006BCD 247F              <1>       and   al, 07Fh            ; ENSURE SET BIT TURNED OFF
00006BCF 0C20              <1>       or    al, 20h                   ; TURN ON ALARM ENABLE
00006BD1 86E0              <1>       xchg  ah, al              ; MOVE MASK TO OUTPUT REGISTER
00006BD3 E8EC000000        <1>       call  CMOS_WRITE          ; WRITE NEW ALARM MASK
00006BD8 F8                <1>       clc                       ; SET CY= 0
```

```
                        <1> RTC_69:
00006BD9 66B80000       <1>       mov    ax, 0               ; CLEAR AX REGISTER
00006BDD C3             <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
00006BDE 66B80B0B       <1>       mov    ax, CMOS_REG_B * 257       ; ADDRESS ALARM REGISTER (TO BOTH AH,AL)
00006BE2 E8C4000000     <1>       call   CMOS_READ          ; READ ALARM REGISTER
00006BE7 2457           <1>       and    al, 57h                   ; TURN OFF ALARM ENABLE
00006BE9 86E0           <1>       xchg   ah, al             ; SAVE DATA AND RECOVER ADDRESS
00006BEB E8D4000000     <1>       call   CMOS_WRITE         ; RESTORE NEW VALUE
00006BF0 F8             <1>       clc                       ; SET CY= 0
00006BF1 C3             <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
00006BF2 66B80A26       <1>       mov    ax, (26h*100h)+CMOS_REG_A
00006BF6 E8C9000000     <1>       call   CMOS_WRITE         ; INITIALIZE STATUS REGISTER A
                        <1>       ;mov   al, CMOS_REG_B             ; SET "SET BIT" FOR CLOCK INITIALIZATION
                        <1>       ;mov   ah, 82h
00006BFB 66B80B82       <1>       mov    ax, (82h*100h)+CMOS_REG_B
00006BFF E8C0000000     <1>       call   CMOS_WRITE         ; AND 24 HOUR MODE TO REGISTER B
00006C04 B00C           <1>       mov    al, CMOS_REG_C             ; ADDRESS REGISTER C
00006C06 E8A0000000     <1>       call   CMOS_READ          ; READ REGISTER C TO INITIALIZE
00006C0B B00D           <1>       mov    al, CMOS_REG_D             ; ADDRESS REGISTER D
00006C0D E899000000     <1>       call   CMOS_READ          ; READ REGISTER D TO INITIALIZE
00006C12 C3             <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
00006C13 1E             <1>       push   ds                  ; LEAVE INTERRUPTS DISABLED
00006C14 50             <1>       push   eax                 ; SAVE REGISTERS
00006C15 57             <1>       push   edi
                        <1> RTC_I_1:                         ; CHECK FOR SECOND INTERRUPT
00006C16 66B88C8B       <1>       mov    ax, 256*(CMOS_REG_B+NMI)+CMOS_REG_C+NMI ; ALARM AND STATUS
00006C1A E670           <1>       out    CMOS_PORT, al       ; WRITE ALARM FLAG MASK ADDRESS
00006C1C 90             <1>       nop                        ; I/O DELAY
00006C1D EB00           <1>       jmp    short $+2
00006C1F E471           <1>       in     al, CMOS_DATA       ; READ AND RESET INTERRUPT REQUEST FLAGS
00006C21 A860           <1>       test   al, 01100000b       ; CHECK FOR EITHER INTERRUPT PENDING
00006C23 745D           <1>       jz     short  RTC_I_9            ; EXIT IF NOT A VALID RTC INTERRUPT
                        <1>
00006C25 86E0           <1>       xchg   ah, al              ; SAVE FLAGS AND GET ENABLE ADDRESS
00006C27 E670           <1>       out    CMOS_PORT, al       ; WRITE ALARM ENABLE MASK ADDRESS
00006C29 90             <1>       nop                        ; I/O DELAY
00006C2A EB00           <1>       jmp    short $+2
00006C2C E471           <1>       in     al, CMOS_DATA       ; READ CURRENT ALARM ENABLE MASK
00006C2E 20E0           <1>       and    al, ah              ; ALLOW ONLY SOURCES THAT ARE ENABLED
00006C30 A840           <1>       test   al, 01000000b       ; CHECK FOR PERIODIC INTERRUPT
00006C32 743B           <1>       jz     short RTC_I_5       ; SKIP IF NOT A PERIODIC INTERRUPT
                        <1>
                        <1> ;-----       DECREMENT WAIT COUNT BY INTERRUPT INTERVAL
                        <1>
00006C34 66BF1000       <1>       mov    di, KDATA           ; kernel data segment
00006C38 8EDF           <1>       mov    ds, di
                        <1>
00006C3A 812D[08820100]D003- <1>  sub    dword [RTC_LH], 976 ; DECREMENT COUNT BY 1/1024
00006C42 0000           <1>
00006C44 7329           <1>       jnc    short RTC_I_5       ; SKIP TILL 32 BIT WORD LESS THAN ZERO
                        <1>
                        <1> ;-----       TURN OFF PERIODIC INTERRUPT ENABLE
                        <1>
00006C46 6650           <1>       push   ax                  ; SAVE INTERRUPT FLAG MASK
00006C48 66B88B8B       <1>       mov    ax, 257*(CMOS_REG_B+NMI) ; INTERRUPT ENABLE REGISTER
00006C4C E670           <1>       out    CMOS_PORT, al       ; WRITE ADDRESS TO CMOS CLOCK
00006C4E 90             <1>       nop                        ; I/O DELAY
00006C4F EB00           <1>       jmp    short $+2
00006C51 E471           <1>       in     al, CMOS_DATA       ; READ CURRENT ENABLES
00006C53 24BF           <1>       and    al, 0BFh            ; TURN OFF PIE
00006C55 86C4           <1>       xchg   al, ah              ; GET CMOS ADDRESS AND SAVE VALUE
00006C57 E670           <1>       out    CMOS_PORT, al       ; ADDRESS REGISTER B
00006C59 86C4           <1>       xchg   al, ah              ; GET NEW INTERRUPT ENABLE MASK
00006C5B E671           <1>       out    CMOS_DATA, al       ; SET MASK IN INTERRUPT ENABLE REGISTER
00006C5D C605[0C820100]00    <1>  mov    byte [RTC_WAIT_FLAG], 0   ; SET FUNCTION ACTIVE FLAG OFF
00006C64 8B3D[0D820100]      <1>  mov    edi, [USER_FLAG]    ; SET UP (DS:DI) TO POINT TO USER FLAG
00006C6A C60780              <1>  mov    byte [edi], 80h           ; TURN ON USERS FLAG
00006C6D 6658           <1>       pop    ax                  ; GET INTERRUPT SOURCE BACK
                        <1> RTC_I_5:
00006C6F A820           <1>       test   al, 00100000b       ; TEST FOR ALARM INTERRUPT
00006C71 740D           <1>       jz     short RTC_I_7       ; SKIP USER INTERRUPT CALL IF NOT ALARM
                        <1>
00006C73 B00D           <1>       mov    al, CMOS_REG_D             ; POINT TO DEFAULT READ ONLY REGISTER
00006C75 E670           <1>       out    CMOS_PORT, al       ; ENABLE NMI AND CMOS ADDRESS TO DEFAULT
00006C77 FB             <1>       sti                        ; INTERRUPTS BACK ON NOW
00006C78 52             <1>       push   edx
00006C79 E89CB80000     <1>       call   INT4Ah              ; TRANSFER TO USER ROUTINE
```

```
00006C7E 5A                <1>       pop    edx
00006C7F FA                <1>       cli                         ; BLOCK INTERRUPT FOR RETRY
                           <1> RTC_I_7:                           ; RESTART ROUTINE TO HANDLE DELAYED
00006C80 EB94              <1>       jmp    short RTC_I_1        ;  ENTRY AND SECOND EVENT BEFORE DONE
                           <1>
                           <1> RTC_I_9:                           ; EXIT - NO PENDING INTERRUPTS
00006C82 B00D              <1>       mov    al, CMOS_REG_D              ; POINT TO DEFAULT READ ONLY REGISTER
00006C84 E670              <1>       out    CMOS_PORT, al       ; ENABLE NMI AND CMOS ADDRESS TO DEFAULT
00006C86 B020              <1>       mov    al, EOI                     ; END OF INTERRUPT MASK TO 8259 - 2
00006C88 E6A0              <1>       out    INTB00, al          ; TO 8259 - 2
00006C8A E620              <1>       out    INTA00,      al             ; TO 8259 - 1
00006C8C 5F                <1>       pop    edi                 ; RESTORE REGISTERS
00006C8D 58                <1>       pop    eax
00006C8E 1F                <1>       pop    ds
00006C8F 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
00006C90 51                <1>       push   ecx
                           <1>
                           <1>       ; 29/05/2016
00006C91 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:
00006C96 B00A              <1>       mov    al, CMOS_REG_A              ; ADDRESS STATUS REGISTER A
00006C98 FA                <1>       cli                         ; NO TIMER INTERRUPTS DURING UPDATES
00006C99 E80D000000        <1>       call   CMOS_READ           ; READ UPDATE IN PROCESS FLAG
00006C9E A880              <1>       test   al, 80h                   ; IF UIP BIT IS ON ( CANNOT READ TIME )
00006CA0 7406              <1>       jz     short UPD_90        ; EXIT WITH CY= 0 IF CAN READ CLOCK NOW
00006CA2 FB                <1>       sti                        ; ALLOW INTERRUPTS WHILE WAITING
00006CA3 E2F1              <1>       loop   UPD_10              ; LOOP TILL READY OR TIMEOUT
00006CA5 31C0              <1>       xor    eax, eax            ; CLEAR RESULTS IF ERROR
                           <1>             ; xor ax, ax
00006CA7 F9                <1>       stc                        ; SET CARRY FOR ERROR
                           <1> UPD_90:
00006CA8 59                <1>       pop    ecx                 ; RESTORE CALLERS REGISTER
00006CA9 FA                <1>       cli                        ; INTERRUPTS OFF DURING SET
00006CAA 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:
00006CAB 9C                <1>       pushf                      ; SAVE INTERRUPT ENABLE STATUS AND FLAGS
00006CAC D0C0              <1>       rol    al, 1               ; MOVE NMI BIT TO LOW POSITION
00006CAE F9                <1>       stc                        ; FORCE NMI BIT ON IN CARRY FLAG
00006CAF D0D8              <1>       rcr    al, 1               ; HIGH BIT ON TO DISABLE NMI - OLD IN CY
00006CB1 FA                <1>       cli                        ; DISABLE INTERRUPTS
00006CB2 E670              <1>       out    CMOS_PORT, al       ; ADDRESS LOCATION AND DISABLE NMI
                           <1>       ; 29/05/2016
                           <1>       ;nop                       ; I/O DELAY
00006CB4 E6EB              <1>       out    0ebh,al      ; NEWIODELAY ; AWARD BIOS 1999, ATIME.ASM
                           <1>       ;
00006CB6 E471              <1>       in     al, CMOS_DATA       ; READ THE REQUESTED CMOS LOCATION
00006CB8 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)
00006CBA 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
00006CBC D0D8              <1>       rcr    al, 1               ; PUT ORIGINAL NMI MASK BIT INTO ADDRESS
00006CBE E670              <1>       out    CMOS_PORT, al       ; SET DEFAULT TO READ ONLY REGISTER
00006CC0 6658              <1>       pop    ax                  ; RESTORE (AH) AND (AL), CMOS BYTE
00006CC2 9D                <1>       popf
00006CC3 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)
00006CC4 9C                     <1>       pushf                     ; SAVE INTERRUPT ENABLE STATUS AND FLAGS
00006CC5 6650                   <1>       push   ax                 ; SAVE WORK REGISTER VALUES
00006CC7 D0C0                   <1>       rol    al, 1              ; MOVE NMI BIT TO LOW POSITION
00006CC9 F9                     <1>       stc                       ; FORCE NMI BIT ON IN CARRY FLAG
00006CCA D0D8                   <1>       rcr    al, 1              ; HIGH BIT ON TO DISABLE NMI - OLD IN CY
00006CCC FA                     <1>       cli                       ; DISABLE INTERRUPTS
00006CCD E670                   <1>       out    CMOS_PORT, al      ; ADDRESS LOCATION AND DISABLE NMI
00006CCF 88E0                   <1>       mov    al, ah             ; GET THE DATA BYTE TO WRITE
00006CD1 E671                   <1>       out    CMOS_DATA, al      ; PLACE IN REQUESTED CMOS LOCATION
00006CD3 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
00006CD5 D0D8                   <1>       rcr    al, 1              ; PUT ORIGINAL NMI MASK BIT INTO ADDRESS
00006CD7 E670                   <1>       out    CMOS_PORT, al      ; SET DEFAULT TO READ ONLY REGISTER
00006CD9 90                     <1>       nop                       ; I/O DELAY
00006CDA E471                   <1>       in     al, CMOS_DATA      ; OPEN STANDBY LATCH
00006CDC 6658                   <1>       pop    ax                 ; RESTORE WORK REGISTERS
00006CDE 9D                     <1>       popf
00006CDF C3                     <1>       retn
                                <1>
                                <1> ; /// End Of TIMER FUNCTIONS ///

                                    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)
                                          ;
00006CE0 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
00006CE8 FFFF0000009ACF00                 dw 0FFFFh, 0, 9A00h, 00CFh; KCODE
                                    gdt_kdata:
                                                              ; 10h kernel data segment, base = 00000000h
00006CF0 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
00006CF8 FFFB000040FACF00                 dw 0FBFFh, 0, 0FA40h, 00CFh      ; UCODE
                                    gdt_udata:
                                                              ; 23h user data segment, base address = 400000h ; CORE
00006D00 FFFB000040F2CF00                 dw 0FBFFh, 0, 0F240h, 00CFh      ; UDATA
                                    gdt_tss:
                                                              ; Task State Segment
00006D08 6700                             dw 0067h ; Limit = 103 ; (104-1, tss size = 104 byte,
                                                                   ;  no IO permission in ring 3)
                                    gdt_tss0:
00006D0A 0000                             dw 0  ; TSS base address, bits 0-15
                                    gdt_tss1:
00006D0C 00                               db 0  ; TSS base address, bits 16-23
                                                              ; 49h
00006D0D E9                               db 11101001b ; 0E9h => P=1/DPL=11/0/1/0/B/1 --> B = Task is busy (1)
00006D0E 00                               db 0 ; G/0/0/AVL/LIMIT=0000 ; (Limit bits 16-19 = 0000) (G=0, 1 byte)
                                    gdt_tss2:
00006D0F 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
00006D10 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
00006D18 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
00006D20 FFFF00000A920000                 dw 0FFFFh, 0, 920Ah, 0
                                                              ; 48h ; VBE3B000
                                    _B0000Sel: ; MDA (monochrome) video memory address
                                          ; limit = 65536, base addr = 0B0000h, 16 bit
00006D28 FFFF00000B920000                 dw 0FFFFh, 0, 920Bh, 0
                                                              ; 50h ; VBE3B800
                                    _B8000Sel: ; CGA video memory address
                                          ; limit = 32768, base addr = 0B8000h, 16 bit
00006D30 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
00006D38 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
00006D40 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
00006D48 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
00006D50 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:
00006D58 7700                           dw gdt_end - gdt - 1    ; Limit (size)
00006D5A [E06C0000]                     dd gdt                  ; Address of the GDT
                                
                                     ; 20/08/2014
                                idtd:
00006D5E 7F02                           dw idt_end - idt - 1    ; Limit (size)
00006D60 [A87E0100]                     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>
00006D64 [C76D0000]           <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:
00006D68 01             <1>              DB     01           ;DRIVE TYPE, MEDIA TABLE
                        <1>                 ;DW      MD_TBL1
00006D69 [866D0000]     <1>              dd     MD_TBL1
00006D6D 82             <1>              DB     02+BIT7ON
                        <1>              ;DW      MD_TBL2
00006D6E [936D0000]     <1>                 dd      MD_TBL2
00006D72 02             <1> DR_DEFAULT:  DB     02
                        <1>                 ;DW      MD_TBL3
00006D73 [A06D0000]     <1>              dd      MD_TBL3
00006D77 03             <1>              DB     03
                        <1>                 ;DW      MD_TBL4
00006D78 [AD6D0000]     <1>              dd      MD_TBL4
00006D7C 84             <1>              DB     04+BIT7ON
                        <1>                 ;DW      MD_TBL5
00006D7D [BA6D0000]     <1>              dd      MD_TBL5
00006D81 04             <1>              DB     04
                        <1>                 ;DW      MD_TBL6
00006D82 [C76D0000]     <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:
00006D86 DF             <1>              DB     11011111B    ; SRT=D, HD UNLOAD=0F - 1ST SPECIFY BYTE
00006D87 02             <1>              DB     2            ; HD LOAD=1, MODE=DMA - 2ND SPECIFY BYTE
00006D88 25             <1>              DB     MOTOR_WAIT   ; WAIT TIME AFTER OPERATION TILL MOTOR OFF
00006D89 02             <1>              DB     2            ; 512 BYTES/SECTOR
00006D8A 09             <1>              DB     09           ; EOT (LAST SECTOR ON TRACK)
00006D8B 2A             <1>              DB     02AH         ; GAP LENGTH
00006D8C FF             <1>              DB     0FFH         ; DTL
00006D8D 50             <1>              DB     050H         ; GAP LENGTH FOR FORMAT
00006D8E F6             <1>              DB     0F6H         ; FILL BYTE FOR FORMAT
00006D8F 0F             <1>              DB     15           ; HEAD SETTLE TIME (MILLISECONDS)
00006D90 08             <1>              DB     8            ; MOTOR START TIME (1/8 SECONDS)
00006D91 27             <1>              DB     39           ; MAX. TRACK NUMBER
00006D92 80             <1>              DB     RATE_250     ; DATA TRANSFER RATE
                        <1> ;--------------------------------------------------------
                        <1> ;     360 KB MEDIA IN 1.2 MB DRIVE                 :
                        <1> ;--------------------------------------------------------
                        <1> MD_TBL2:
00006D93 DF             <1>              DB     11011111B    ; SRT=D, HD UNLOAD=0F - 1ST SPECIFY BYTE
```

```
00006D94 02                     <1>       DB     2             ; HD LOAD=1, MODE=DMA - 2ND SPECIFY BYTE
00006D95 25                     <1>       DB     MOTOR_WAIT    ; WAIT TIME AFTER OPERATION TILL MOTOR OFF
00006D96 02                     <1>       DB     2             ; 512 BYTES/SECTOR
00006D97 09                     <1>       DB     09            ; EOT (LAST SECTOR ON TRACK)
00006D98 2A                     <1>       DB     02AH          ; GAP LENGTH
00006D99 FF                     <1>       DB     0FFH          ; DTL
00006D9A 50                     <1>       DB     050H          ; GAP LENGTH FOR FORMAT
00006D9B F6                     <1>       DB     0F6H          ; FILL BYTE FOR FORMAT
00006D9C 0F                     <1>       DB     15            ; HEAD SETTLE TIME (MILLISECONDS)
00006D9D 08                     <1>       DB     8             ; MOTOR START TIME (1/8 SECONDS)
00006D9E 27                     <1>       DB     39            ; MAX. TRACK NUMBER
00006D9F 40                     <1>       DB     RATE_300      ; DATA TRANSFER RATE
                                <1> ;--------------------------------------------------------
                                <1> ;     1.2 MB MEDIA IN 1.2 MB DRIVE                 :
                                <1> ;--------------------------------------------------------
                                <1> MD_TBL3:
00006DA0 DF                     <1>       DB     11011111B     ; SRT=D, HD UNLOAD=0F - 1ST SPECIFY BYTE
00006DA1 02                     <1>       DB     2             ; HD LOAD=1, MODE=DMA - 2ND SPECIFY BYTE
00006DA2 25                     <1>       DB     MOTOR_WAIT    ; WAIT TIME AFTER OPERATION TILL MOTOR OFF
00006DA3 02                     <1>       DB     2             ; 512 BYTES/SECTOR
00006DA4 0F                     <1>       DB     15            ; EOT (LAST SECTOR ON TRACK)
00006DA5 1B                     <1>       DB     01BH          ; GAP LENGTH
00006DA6 FF                     <1>       DB     0FFH          ; DTL
00006DA7 54                     <1>       DB     054H          ; GAP LENGTH FOR FORMAT
00006DA8 F6                     <1>       DB     0F6H          ; FILL BYTE FOR FORMAT
00006DA9 0F                     <1>       DB     15            ; HEAD SETTLE TIME (MILLISECONDS)
00006DAA 08                     <1>       DB     8             ; MOTOR START TIME (1/8 SECONDS)
00006DAB 4F                     <1>       DB     79            ; MAX. TRACK NUMBER
00006DAC 00                     <1>       DB     RATE_500      ; DATA TRANSFER RATE
                                <1> ;--------------------------------------------------------
                                <1> ;     720 KB MEDIA IN 720 KB DRIVE                 :
                                <1> ;--------------------------------------------------------
                                <1> MD_TBL4:
00006DAD DF                     <1>       DB     11011111B     ; SRT=D, HD UNLOAD=0F - 1ST SPECIFY BYTE
00006DAE 02                     <1>       DB     2             ; HD LOAD=1, MODE=DMA - 2ND SPECIFY BYTE
00006DAF 25                     <1>       DB     MOTOR_WAIT    ; WAIT TIME AFTER OPERATION TILL MOTOR OFF
00006DB0 02                     <1>       DB     2             ; 512 BYTES/SECTOR
00006DB1 09                     <1>       DB     09            ; EOT (LAST SECTOR ON TRACK)
00006DB2 2A                     <1>       DB     02AH          ; GAP LENGTH
00006DB3 FF                     <1>       DB     0FFH          ; DTL
00006DB4 50                     <1>       DB     050H          ; GAP LENGTH FOR FORMAT
00006DB5 F6                     <1>       DB     0F6H          ; FILL BYTE FOR FORMAT
00006DB6 0F                     <1>       DB     15            ; HEAD SETTLE TIME (MILLISECONDS)
00006DB7 08                     <1>       DB     8             ; MOTOR START TIME (1/8 SECONDS)
00006DB8 4F                     <1>       DB     79            ; MAX. TRACK NUMBER
00006DB9 80                     <1>       DB     RATE_250      ; DATA TRANSFER RATE
                                <1> ;--------------------------------------------------------
                                <1> ;     720 KB MEDIA IN 1.44 MB DRIVE                :
                                <1> ;--------------------------------------------------------
                                <1> MD_TBL5:
00006DBA DF                     <1>       DB     11011111B     ; SRT=D, HD UNLOAD=0F - 1ST SPECIFY BYTE
00006DBB 02                     <1>       DB     2             ; HD LOAD=1, MODE=DMA - 2ND SPECIFY BYTE
00006DBC 25                     <1>       DB     MOTOR_WAIT    ; WAIT TIME AFTER OPERATION TILL MOTOR OFF
00006DBD 02                     <1>       DB     2             ; 512 BYTES/SECTOR
00006DBE 09                     <1>       DB     09            ; EOT (LAST SECTOR ON TRACK)
00006DBF 2A                     <1>       DB     02AH          ; GAP LENGTH
00006DC0 FF                     <1>       DB     0FFH          ; DTL
00006DC1 50                     <1>       DB     050H          ; GAP LENGTH FOR FORMAT
00006DC2 F6                     <1>       DB     0F6H          ; FILL BYTE FOR FORMAT
00006DC3 0F                     <1>       DB     15            ; HEAD SETTLE TIME (MILLISECONDS)
00006DC4 08                     <1>       DB     8             ; MOTOR START TIME (1/8 SECONDS)
00006DC5 4F                     <1>       DB     79            ; MAX. TRACK NUMBER
00006DC6 80                     <1>       DB     RATE_250      ; DATA TRANSFER RATE
                                <1> ;--------------------------------------------------------
                                <1> ;     1.44 MB MEDIA IN 1.44 MB DRIVE               :
                                <1> ;--------------------------------------------------------
                                <1> MD_TBL6:
00006DC7 AF                     <1>       DB     10101111B     ; SRT=A, HD UNLOAD=0F - 1ST SPECIFY BYTE
00006DC8 02                     <1>       DB     2             ; HD LOAD=1, MODE=DMA - 2ND SPECIFY BYTE
00006DC9 25                     <1>       DB     MOTOR_WAIT    ; WAIT TIME AFTER OPERATION TILL MOTOR OFF
00006DCA 02                     <1>       DB     2             ; 512 BYTES/SECTOR
00006DCB 12                     <1>       DB     18            ; EOT (LAST SECTOR ON TRACK)
00006DCC 1B                     <1>       DB     01BH          ; GAP LENGTH
00006DCD FF                     <1>       DB     0FFH          ; DTL
00006DCE 6C                     <1>       DB     06CH          ; GAP LENGTH FOR FORMAT
00006DCF F6                     <1>       DB     0F6H          ; FILL BYTE FOR FORMAT
00006DD0 0F                     <1>       DB     15            ; HEAD SETTLE TIME (MILLISECONDS)
00006DD1 08                     <1>       DB     8             ; MOTOR START TIME (1/8 SECONDS)
00006DD2 4F                     <1>       DB     79            ; MAX. TRACK NUMBER
00006DD3 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:
00006DD4 E0             <1>       db     NO_ERR
00006DD5 024001BB       <1>       db     BAD_ADDR_MARK,BAD_SEEK,BAD_CMD,UNDEF_ERR
00006DD9 04BB100A       <1>       db     RECORD_NOT_FND,UNDEF_ERR,BAD_ECC,BAD_SECTOR
                        <1>
                        <1> ; 17/12/2014 (mov ax, [cfd])
                        <1> ; 11/12/2014
00006DDD 00             <1> cfd:         db 0                ; current floppy drive (for GET_PARM)
                        <1> ; 17/12/2014                     ; instead of 'DISK_POINTER'
00006DDE 01             <1> pfd:         db 1                ; previous floppy drive (for GET_PARM)
                        <1>                                  ; (initial value of 'pfd
                        <1>                                  ; must be different then 'cfd' value
                        <1>                                  ; to force updating/initializing
                        <1>                                  ; current drive parameters)
00006DDF 90             <1> align 2
                        <1>
00006DE0 F001           <1> HF_PORT:     dw     1F0h  ; Default = 1F0h
                        <1>                            ; (170h)
00006DE2 F603           <1> HF_REG_PORT: dw     3F6h  ; HF_PORT + 206h
                        <1>
                        <1> ; 05/01/2015
00006DE4 00             <1> hf_m_s:         db      0     ; (0 = Master, 1 = Slave)
                        <1>
                        <1> ; ****************************************************************************
                            
                            Align 2
                            
                            ; 04/11/2014 (Retro UNIX 386 v1)
00006DE6 0000               mem_1m_1k:   dw 0  ; Number of contiguous KB between
                                               ; 1 and 16 MB, max. 3C00h = 15 MB.
00006DE8 0000               mem_16m_64k: dw 0  ; Number of contiguous 64 KB blocks
                                           ; between 16 MB and 4 GB.
                            
                            ; 12/11/2014 (Retro UNIX 386 v1)
00006DEA 00                 boot_drv:    db 0 ; boot drive number (physical)
                            ; 24/11/2014
00006DEB 00                 drv:       db 0
00006DEC 00                 last_drv:    db 0 ; last hdd
00006DED 00                 hdc:         db 0  ; number of hard disk drives
                                             ; (present/detected)
                            
                            ; 24/11/2014 (Retro UNIX 386 v1)
                            ; Physical drive type & flags
00006DEE 00                 fd0_type:    db 0  ; floppy drive type
00006DEF 00                 fd1_type:    db 0  ; 4 = 1.44 Mb, 80 track, 3.5" (18 spt)
                                             ; 6 = 2.88 Mb, 80 track, 3.5" (36 spt)
                                             ; 3 = 720 Kb, 80 track, 3.5" (9 spt)
                                             ; 2 = 1.2 Mb, 80 track, 5.25" (15 spt)
                                             ; 1 = 360 Kb, 40 track, 5.25" (9 spt)
00006DF0 00                 hd0_type:    db 0  ; EDD status for hd0 (bit 7 = present flag)
00006DF1 00                 hd1_type:    db 0  ; EDD status for hd1 (bit 7 = present flag)
00006DF2 00                 hd2_type:    db 0  ; EDD status for hd2 (bit 7 = present flag)
00006DF3 00                 hd3_type:    db 0  ; EDD status for hd3 (bit 7 = present flag)
                                             ; bit 0 - Fixed disk access subset supported
                                             ; bit 1 - Drive locking and ejecting
                                             ; bit 2 - Enhanced disk drive support
                                             ; bit 3 - Reserved (64 bit EDD support)
                                             ; (If bit 0 is '1' Retro UNIX 386 v1
                                             ; will interpret it as 'LBA ready'!)
                            
                                ; 11/03/2015 - 10/07/2015
00006DF4 00000000000000000- drv.cylinders: dw 0,0,0,0,0,0,0
00006DFD 0000000000
00006E02 00000000000000000- drv.heads:     dw 0,0,0,0,0,0,0
00006E0B 0000000000
00006E10 00000000000000000- drv.spt:       dw 0,0,0,0,0,0,0
00006E19 0000000000
00006E1E 00000000000000000- drv.size:      dd 0,0,0,0,0,0,0
00006E27 00000000000000000-
00006E30 00000000000000000-
00006E39 00
00006E3A 00000000000000         drv.status:    db 0,0,0,0,0,0,0
00006E41 00000000000000         drv.error:     db 0,0,0,0,0,0,0
                            
                                Align 2
                            
                                ;;; 11/03/2015
                                %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> ;
```

```
                        <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
                        <1> ; kybdata.inc (11/03/2015)
                        <1> ;
                        <1> ; Derived from 'IBM PC-XT-286' BIOS source code (1986)
                        <1> ; ****************************************************************************
                        <1>
                        <1> ; Retro UNIX 386 v1 Kernel - KYBDATA.INC
                        <1> ; Last Modification: 11/03/2015
                        <1> ;             (Data Section for 'KEYBOARD.INC')
                        <1> ;
                        <1> ; ///////// KEYBOARD DATA ////////////////
                        <1>
                        <1> ; 05/12/2014
                        <1> ; 04/12/2014 (derived from pc-xt-286 bios source code -1986-)
                        <1> ; 03/06/86  KEYBOARD BIOS
                        <1>
                        <1> ;--------------------------------------------------------------------------------
                        <1> ;     KEY IDENTIFICATION SCAN TABLES
                        <1> ;--------------------------------------------------------------------------------
                        <1>
                        <1> ;-----      TABLES FOR ALT CASE ------------
                        <1> ;-----      ALT-INPUT-TABLE
00006E48 524F50514B     <1> K30:  db    82,79,80,81,75
00006E4D 4C4D474849     <1>       db    76,77,71,72,73            ; 10 NUMBER ON KEYPAD
                        <1> ;-----      SUPER-SHIFT-TABLE
00006E52 101112131415   <1>       db    16,17,18,19,20,21   ; A-Z TYPEWRITER CHARS
00006E58 161718191E1F   <1>       db    22,23,24,25,30,31
00006E5E 202122232425   <1>       db    32,33,34,35,36,37
00006E64 262C2D2E2F30   <1>       db    38,44,45,46,47,48
00006E6A 3132           <1>       db    49,50
                        <1>
                        <1> ;-----      TABLE OF SHIFT KEYS AND MASK VALUES
                        <1> ;-----      KEY TABLE
00006E6C 52             <1> _K6:    db      INS_KEY                 ; INSERT KEY
00006E6D 3A4546381D     <1>         db    CAPS_KEY,NUM_KEY,SCROLL_KEY,ALT_KEY,CTL_KEY
00006E72 2A36           <1>         db      LEFT_KEY,RIGHT_KEY
                        <1> _K6L    equ     $-_K6
                        <1>
                        <1> ;-----      MASK_TABLE
00006E74 80             <1> _K7:    db      INS_SHIFT               ; INSERT MODE SHIFT
00006E75 4020100804     <1>         db    CAPS_SHIFT,NUM_SHIFT,SCROLL_SHIFT,ALT_SHIFT,CTL_SHIFT
00006E7A 0201           <1>         db    LEFT_SHIFT,RIGHT_SHIFT
                        <1>
                        <1> ;-----      TABLES FOR CTRL CASE            ;---- CHARACTERS ------
00006E7C 1BFF00FFFFFF   <1> _K8:  db    27,-1,0,-1,-1,-1    ; Esc, 1, 2, 3, 4, 5
00006E82 1EFFFFFFFF1F   <1>       db    30,-1,-1,-1,-1,31   ; 6, 7, 8, 9, 0, -
00006E88 FF7FFF111705   <1>       db    -1,127,-1,17,23,5   ; =, Bksp, Tab, Q, W, E
00006E8E 12141915090F   <1>       db    18,20,25,21,9,15    ; R, T, Y, U, I, O
00006E94 101B1D0AFF01   <1>       db    16,27,29,10,-1,1    ; P, [, ], Enter, Ctrl, A
00006E9A 13040607080A   <1>       db    19,4,6,7,8,10       ; S, D, F, G, H, J
00006EA0 0B0CFFFFFFFF   <1>       db    11,12,-1,-1,-1,-1   ; K, L, :, ', `, LShift
00006EA6 1C1A18031602   <1>       db    28,26,24,3,22,2           ; Bkslash, Z, X, C, V, B
00006EAC 0E0DFFFFFFFF   <1>       db    14,13,-1,-1,-1,-1   ; N, M, ,, ., /, RShift
00006EB2 96FF20FF       <1>       db    150,-1,' ',-1       ; *, ALT, Spc, CL
                        <1>       ;                         ;----- FUNCTIONS ------
00006EB6 5E5F60616263   <1>       db    94,95,96,97,98,99   ; F1 - F6
00006EBC 64656667FFFF   <1>       db    100,101,102,103,-1,-1     ; F7 - F10, NL, SL
00006EC2 778D848E738F   <1>       db    119,141,132,142,115,143   ; Home, Up, PgUp, -, Left, Pad5
00006EC8 749075917692   <1>       db    116,144,117,145,118,146 ; Right, +, End, Down, PgDn, Ins
00006ECE 93FFFFFF898A   <1>       db    147,-1,-1,-1,137,138      ; Del, SysReq, Undef, WT, F11, F12
                        <1>
                        <1> ;-----      TABLES FOR LOWER CASE ----------
00006ED4 1B3132333435363738- <1> K10:  db    27,'1234567890-=',8,9
00006EDD 39302D3D0809   <1>
00006EE3 7177657274797569 6F- <1>       db    'qwertyuiop[]',13,-1,'asdfghjkl;',39
00006EEC 705B5D0DFF61736466- <1>
00006EF5 67686A6B6C3B27 <1>
00006EFC 60FF5C7A786376626E- <1>       db    96,-1,92,'zxcvbnm,./',-1,'*',-1,' ',-1
00006F05 6D2C2E2FFF2AFF20FF  <1>
                        <1> ;-----      LC TABLE SCAN
00006F0E 3B3C3D3E3F     <1>       db    59,60,61,62,63            ; BASE STATE OF F1 - F10
00006F13 4041424344     <1>       db    64,65,66,67,68
00006F18 FFFF           <1>       db    -1,-1               ; NL, SL
                        <1>
                        <1> ;-----      KEYPAD TABLE
00006F1A 474849FF4BFF   <1> K15:  db    71,72,73,-1,75,-1   ; BASE STATE OF KEYPAD KEYS
00006F20 4DFF4F50515253 <1>       db    77,-1,79,80,81,82,83
00006F27 FFFF5C8586     <1>       db    -1,-1,92,133,134    ; SysRq, Undef, WT, F11, F12
                        <1>
                        <1> ;-----      TABLES FOR UPPER CASE ----------
00006F2C 1B21402324255E262A- <1> K11:  db    27,'!@#$%',94,'&*()_+',8,0
00006F35 28295F2B0800   <1>
00006F3B 5157455254595549 4F- <1>       db    'QWERTYUIOP{}',13,-1,'ASDFGHJKL:"'
00006F44 507B7D0DFF41534446- <1>
00006F4D 47484A4B4C3A22 <1>
00006F54 7EFF7C5A584356424E- <1>       db    126,-1,'|ZXCVBNM<>?',-1,'*',-1,' ',-1
00006F5D 4D3C3E3FFF2AFF20FF  <1>
                        <1> ;-----      UC TABLE SCAN
00006F66 5455565758     <1> K12:  db    84,85,86,87,88            ; SHIFTED STATE OF F1 - F10
00006F6B 595A5B5C5D     <1>       db    89,90,91,92,93
00006F70 FFFF           <1>       db    -1,-1               ; NL, SL
                        <1>
                        <1> ;-----      NUM STATE TABLE
00006F72 3738392D3435362B31- <1> K14:  db    '789-456+1230.'           ; NUMLOCK STATE OF KEYPAD KEYS
00006F7B 3233302E       <1>
                        <1>       ;
00006F7F FFFF7C8788     <1>       db    -1,-1,124,135,136   ; SysRq, Undef, WT, F11, F12
                        <1>
                        <1> ; 26/08/2014
                        <1> ; Retro UNIX 8086 v1 - UNIX.ASM (03/03/2014)
                        <1> ; Derived from IBM "pc-at"
                        <1> ; rombios source code (06/10/1985)
                        <1> ; 'dseg.inc'
                        <1>
                        <1> ;--------------------------------------;
```

```
                        <1> ;     SYSTEM DATA AREA          ;
                        <1> ;---------------------------------------
00006F84 00             <1> BIOS_BREAK   db     0            ; BIT 7=1 IF BREAK KEY HAS BEEN PRESSED
                        <1>
                        <1> ;---------------------------------------
                        <1> ;     KEYBOARD DATA AREAS       ;
                        <1> ;---------------------------------------
                        <1>
00006F85 00             <1> KB_FLAG             db     0           ; KEYBOARD SHIFT STATE AND STATUS FLAGS
00006F86 00             <1> KB_FLAG_1    db     0            ; SECOND BYTE OF KEYBOARD STATUS
00006F87 00             <1> KB_FLAG_2    db     0            ; KEYBOARD LED FLAGS
00006F88 00             <1> KB_FLAG_3    db     0            ; KEYBOARD MODE STATE AND TYPE FLAGS
00006F89 00             <1> ALT_INPUT    db     0            ; STORAGE FOR ALTERNATE KEY PAD ENTRY
00006F8A [9A6F0000]     <1> BUFFER_START dd     KB_BUFFER    ; OFFSET OF KEYBOARD BUFFER START
00006F8E [BA6F0000]     <1> BUFFER_END   dd     KB_BUFFER + 32      ; OFFSET OF END OF BUFFER
00006F92 [9A6F0000]     <1> BUFFER_HEAD  dd     KB_BUFFER    ; POINTER TO HEAD OF KEYBOARD BUFFER
00006F96 [9A6F0000]     <1> BUFFER_TAIL  dd     KB_BUFFER    ; POINTER TO TAIL OF KEYBOARD BUFFER
                        <1> ; ------     HEAD = TAIL  INDICATES THAT THE BUFFER IS EMPTY
00006F9A 0000<rept>     <1> KB_BUFFER    times  16 dw 0             ; ROOM FOR 16 SCAN CODE ENTRIES
                        <1>
                        <1> ; /// End Of KEYBOARD DATA ///
                            %include 'vidata.s'       ; VIDEO (BIOS) DATA
                        <1> ; ****************************************************************************
                        <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.3 - vidata.s
                        <1> ; ----------------------------------------------------------------------------
                        <1> ; Last Update: 24/11/2020
                        <1> ; ----------------------------------------------------------------------------
                        <1> ; Beginning: 16/01/2016
                        <1> ; ----------------------------------------------------------------------------
                        <1> ; Assembler: NASM version 2.15 (trdos386.s)
                        <1> ; ----------------------------------------------------------------------------
                        <1> ; Turkish Rational DOS
                        <1> ; Operating System Project v2.0 by ERDOGAN TAN (Beginning: 04/01/2016)
                        <1> ;
                        <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
                        <1> ; vidata.inc (11/03/2015)
                        <1> ;
                        <1> ; Derived from 'IBM PC-AT' BIOS source code (1985)
                        <1> ; ****************************************************************************
                        <1>
                        <1> ; Retro UNIX 386 v1 Kernel - VIDATA.S
                        <1> ; Last Modification: 11/03/2015
                        <1> ;                (Data section for 'VIDEO.INC')
                        <1> ;
                        <1> ; ///////// VIDEO DATA ///////////////
                        <1>
                        <1> ;---------------------------------------
                        <1> ;     VIDEO DISPLAY DATA AREA          ;
                        <1> ;---------------------------------------
00006FBA 03             <1> CRT_MODE:    db     3      ; CURRENT DISPLAY MODE (TYPE)
00006FBB 29             <1> CRT_MODE_SET:      db     29h    ; CURRENT SETTING OF THE 3X8 REGISTER
                        <1>                            ; (29h default setting for video mode 3)
                        <1>                            ; Mode Select register Bits
                        <1>                            ;   BIT 0 - 80x25 (1), 40x25 (0)
                        <1>                            ;   BIT 1 - ALPHA (0), 320x200 GRAPHICS (1)
                        <1>                            ;   BIT 2 - COLOR (0), BW (1)
                        <1>                            ;   BIT 3 - Video Sig. ENABLE (1), DISABLE (0)
                        <1>                            ;   BIT 4 - 640x200 B&W Graphics Mode (1)
                        <1>                            ;   BIT 5 - ALPHA mode BLINKING (1)
                        <1>                            ;   BIT 6, 7 - Not Used
                        <1>
                        <1> ; Mode 0 - 2Ch = 101100b  ; 40x25 text, 16 gray colors
                        <1> ; Mode 1 - 28h = 101000b  ; 40x25 text, 16 fore colors, 8 back colors
                        <1> ; Mode 2 - 2Dh = 101101b  ; 80x25 text, 16 gray colors
                        <1> ; Mode 3 - 29h = 101001b  ; 80x25 text, 16 fore color, 8 back color
                        <1> ; Mode 4 - 2Ah = 101010b  ; 320x200 graphics, 4 colors
                        <1> ; Mode 5 - 2Eh = 101110b  ; 320x200 graphics, 4 gray colors
                        <1> ; Mode 6 - 1Eh = 011110b  ; 640x200 graphics, 2 colors
                        <1> ; Mode 7 - 29h = 101001b  ; 80x25 text, black & white colors
                        <1> ; Mode & 37h = Video signal OFF
                        <1>
                        <1> ; 24/06/2016
00006FBC 50             <1> CRT_COLS:    db     80     ; Number of columns
                        <1>
                        <1> ; 01/07/2016
00006FBD 00             <1> CRT_PALETTE: db     0      ; Current palette setting
                        <1>
                        <1> ; 03/07/2016
00006FBE 10             <1> CHAR_HEIGHT: db     16     ; Default character height
00006FBF 60             <1> VGA_VIDEO_CTL:     db     60h    ; ROM BIOS DATA AREA Offset 87h
00006FC0 F9             <1> VGA_SWITCHES:      db     0F9h   ; Feature Bit Switches (the basic screen)
00006FC1 51             <1> VGA_MODESET_CTL: db       051h   ; Basic mode set options (VGA video flags)
                        <1>                            ; ROM BIOS DATA AREA Offset 89h
                        <1>                            ; Bit 7, 4 : Mode
                        <1>                            ;       01 : 400-line mode
                        <1>                            ; Bit 6      : Display switch enabled = 1
                        <1>                            ; Bit 5      : Reserved = 0
                        <1>                            ; Bit 3      : Default palette loading
                        <1>                            ;       disabled = 0
                        <1>                            ; Bit 2 : Color monitor = 0
                        <1>                            ; Bit 1 = Gray scale summing
                        <1>                            ;       disabled = 0
                        <1>                            ; Bit 0 = VGA active = 1
00006FC2 19             <1> VGA_ROWS:    db     25
                        <1>
                        <1> ; 16/01/2016
                        <1> chr_attrib:  ; Character color/attributes for video pages (0 to 7)
00006FC3 0707070707070707   <1>       db     07h, 07h, 07h, 07h, 07h, 07h, 07h, 07h
                        <1> ; 30/01/2016
                        <1> vmode:
00006FCB 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
00006FD3 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>
00006FD5 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)
00006FD6 0302010007040506       <1>         db      03h, 02h, 01h, 00h, 07h, 04h, 05h, 06h
                                <1> vga_g_modes: ; 31/07/2016
00006FDE 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
00006FE6 [46700000]             <1>         dd      vga_mode_03h
00006FEA [46700000]             <1>         dd      vga_mode_03h ; mode 02h -> mode 03h
00006FEE [86700000]             <1>         dd      vga_mode_01h
00006FF2 [86700000]             <1>         dd      vga_mode_01h ; mode 00h -> mode 01h
                                <1>         ;dd     vga_mode_07h
00006FF6 [46700000]             <1>         dd      vga_mode_03h ; mode 07h -> mode 03h
00006FFA [C6700000]             <1>         dd      vga_mode_04h
00006FFE [C6700000]             <1>         dd      vga_mode_04h ; mode 05h -> mode 04h
00007002 [06710000]             <1>         dd      vga_mode_06h
00007006 [46710000]             <1>         dd      vga_mode_13h
0000700A [86710000]             <1>         dd      vga_mode_F0h
0000700E [C6710000]             <1>         dd      vga_mode_12h
00007012 [06720000]             <1>         dd      vga_mode_6Ah
00007016 [46720000]             <1>         dd      vga_mode_0Dh
0000701A [86720000]             <1>         dd      vga_mode_0Eh
0000701E [C6720000]             <1>         dd      vga_mode_10h
00007022 [06730000]             <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
00007026 0000000000020202       <1>         db      CTEXT, CTEXT, CTEXT, CTEXT, MTEXT, CGA, CGA, CGA
                                <1> vga_g_memmodel: ; 31/07/2016
0000702E 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:
00007036 020202020001010103-    <1>         db      2, 2, 2, 2, 0, 1, 1, 1, 3, 3, 2, 2, 1, 1, 2, 2
0000703F 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
00007046 5018100010             <1>         db      80, 24, 16, 00h, 10h ; tw, th-1, ch, slength (5)
0000704B 00030002               <1>         db      00h, 03h, 00h, 02h ; sequ regs (4)
0000704F 67                     <1>         db      67h    ; misc reg (1)
00007050 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
00007058 004F                   <1>         db      00h, 4Fh
                                <1> vga_p_cm_pos equ $ - vga_mode_03h
0000705A 0D0E00000000           <1>         db      0Dh, 0Eh, 00h, 00h, 00h, 00h
00007060 9C8E8F281F96B9A3       <1>         db      9Ch, 8Eh, 8Fh, 28h, 1Fh, 96h, 0B9h, 0A3h
00007068 FF                     <1>         db      0FFh   ; crtc_regs (25)
```

```
00007069 0001020304051407    <1>       db     00h, 01h, 02h, 03h, 04h, 05h, 14h, 07h
00007071 38393A3B3C3D3E3F    <1>       db     38h, 39h, 3Ah, 3Bh, 3Ch, 3Dh, 3Eh, 3Fh
                             <1>       ;db    0Ch, 00h, 0Fh, 08h  ; actl regs (20)
00007079 0C000F00            <1>       db     0Ch, 00h, 0Fh, 00h  ; 19/11/2020
0000707D 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
00007086 2818100008          <1>       db     40, 24, 16, 00h, 08h ; tw, th-1, ch, slength
0000708B 08030002            <1>       db     08h, 03h, 00h, 02h  ; sequ regs
0000708F 67                  <1>       db     67h    ; misc reg
00007090 2D2728902BA0BF1F    <1>       db     2Dh, 27h, 28h, 90h, 2Bh, 0A0h, 0BFh, 1Fh
00007098 004F0D0E00000000    <1>       db     00h, 4Fh, 0Dh, 0Eh, 00h, 00h, 00h, 00h
000070A0 9C8E8F141F96B9A3    <1>       db     9Ch, 8Eh, 8Fh, 14h, 1Fh, 96h, 0B9h, 0A3h
000070A8 FF                  <1>       db     0FFh   ; crtc_regs
000070A9 0001020304051407    <1>       db     00h, 01h, 02h, 03h, 04h, 05h, 14h, 07h
000070B1 38393A3B3C3D3E3F    <1>       db     38h, 39h, 3Ah, 3Bh, 3Ch, 3Dh, 3Eh, 3Fh
                             <1>       ;db    0Ch, 00h, 0Fh, 08h  ; actl regs (20)
000070B9 0C000F00            <1>       db     0Ch, 00h, 0Fh, 00h  ; 19/11/2020
000070BD 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
000070C6 2818080040          <1>       db     40, 24, 8, 00h, 40h ; tw, th-1, ch, slength
000070CB 09030002            <1>       db     09h, 03h, 00h, 02h ; sequ regs
000070CF 63                  <1>       db     63h    ; misc reg
000070D0 2D2728902B80BF1F    <1>       db     2Dh, 27h, 28h, 90h, 2Bh, 80h, 0BFh, 1Fh
000070D8 00C1000000000000    <1>       db     00h, 0C1h, 00h, 00h, 00h, 00h, 00h, 00h
000070E0 9C8E8F140096B9A2    <1>       db     9Ch, 8Eh, 8Fh, 14h, 00h, 96h, 0B9h, 0A2h
000070E8 FF                  <1>       db     0FFh   ; crtc_regs
000070E9 0013151702040607    <1>       db     00h, 13h, 15h, 17h, 02h, 04h, 06h, 07h
000070F1 1011121314151617    <1>       db     10h, 11h, 12h, 13h, 14h, 15h, 16h, 17h
000070F9 01000300            <1>       db     01h, 00h, 03h, 00h ; actl regs
000070FD 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
00007106 5018080040          <1>       db     80, 24, 8, 00h, 40h   ;   tw, th-1, ch, slength
0000710B 01010006            <1>       db     01h, 01h, 00h, 06h ; sequ regs
0000710F 63                  <1>       db     63h    ; misc reg
00007110 5F4F50825480BF1F    <1>       db     5Fh, 4Fh, 50h, 82h, 54h, 80h, 0BFh, 1Fh
00007118 00C1000000000000    <1>       db     00h, 0C1h, 00h, 00h, 00h, 00h, 00h, 00h
00007120 9C8E8F280096B9C2    <1>       db     9Ch, 8Eh, 8Fh, 28h, 00h, 96h, 0B9h, 0C2h
00007128 FF                  <1>       db     0FFh   ; crtc regs
00007129 0017171717171717    <1>       db     00h, 17h, 17h, 17h, 17h, 17h, 17h, 17h
00007131 1717171717171717    <1>       db     17h, 17h, 17h, 17h, 17h, 17h, 17h, 17h
00007139 01000100            <1>       db     01h, 00h, 01, 00h  ; actl regs
0000713D 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
00007146 28180800FA          <1>       db     40, 24, 8, 00h, 0FAh ; tw, th-1, ch, slength (5)
0000714B 010F000E            <1>       db     01h, 0Fh, 00h, 0Eh ; sequ regs (4)
0000714F 63                  <1>       db     63h    ; misc reg (1)
00007150 5F4F50825480BF1F    <1>       db     5Fh, 4Fh, 50h, 82h, 54h, 80h, 0BFh, 1Fh
00007158 0041000000000000    <1>       db     00h, 041h, 00h, 00h, 00h, 00h, 00h, 00h
00007160 9C8E8F284096B9A3    <1>       db     9Ch, 8Eh, 8Fh, 28h, 40h, 96h, 0B9h, 0A3h
00007168 FF                  <1>       db     0FFh   ; crtc regs (25)
00007169 0001020304050607    <1>       db     00h, 01h, 02h, 03h, 04h, 05h, 06h, 07h
00007171 08090A0B0C0D0E0F    <1>       db     08h, 09h, 0Ah, 0Bh, 0Ch, 0Dh, 0Eh, 0Fh
00007179 41000F00            <1>       db     41h, 00h, 0Fh, 00h  ; actl regs (20)
                             <1>       ; 10/11/2020
                             <1>       ;db    41h, 01h, 0Fh, 13h  ; actl regs (20)
0000717D 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
00007186 2818080000          <1>       db     40, 24, 8, 00h, 00h ; tw, th-1, ch, slength
0000718B 010F0006            <1>       db     01h, 0Fh, 00h, 06h ; sequ regs
0000718F E3                  <1>       db     0E3h   ; misc reg
00007190 5F4F508254800D3E    <1>       db     5Fh, 4Fh, 50h, 82h, 54h, 80h, 0Dh, 3Eh
00007198 0041000000000000    <1>       db     00h, 41h, 00h, 00h, 00h, 00h, 00h, 00h
000071A0 EAACDF2800E706E3    <1>       db     0EAh, 0ACh, 0DFh, 28h, 00h, 0E7h, 06h, 0E3h
000071A8 FF                  <1>       db     0FFh   ; crtc regs (25)
000071A9 0001020304050607    <1>       db     00h, 01h, 02h, 03h, 04h, 05h, 06h, 07h
000071B1 08090A0B0C0D0E0F    <1>       db     08h, 09h, 0Ah, 0Bh, 0Ch, 0Dh, 0Eh, 0Fh
000071B9 41000F00            <1>       db     41h, 00h, 0Fh, 00h  ; actl regs
000071BD 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
000071C6 501D1000A0          <1>       db     80, 29, 16, 00h, 0A0h ; tw, th-1, ch, slength
000071CB 010F0006            <1>       db     01h, 0Fh, 00h, 06h ; sequ regs
000071CF E3                  <1>       db     0E3h   ; misc reg
000071D0 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
000071D8 0040000000000000    <1>       db     00h, 40h, 00h, 00h, 00h, 00h, 00h, 00h
000071E0 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
000071E8 FF                  <1>       db     0FFh   ; crtc regs
000071E9 0001020304051407    <1>       db     00h, 01h, 02h, 03h, 04h, 05h, 14h, 07h
000071F1 38393A3B3C3D3E3F    <1>       db     38h, 39h, 3Ah, 3Bh, 3Ch, 3Dh, 3Eh, 3Fh
```

```
000071F9 01000F00            <1>       db     01h, 00h, 0Fh, 00h  ; actl regs
000071FD 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
00007206 6424100000          <1>       db     100, 36, 16, 00h, 00h ; tw, th-1, ch, slength
0000720B 010F0006            <1>       db     01h, 0Fh, 00h, 06h ; sequ regs
0000720F E3                  <1>       db     0E3h   ; misc reg
00007210 7F6363836B1B72F0    <1>       db     7Fh, 63h, 63h, 83h, 6Bh, 1Bh, 72h, 0F0h
00007218 0060000000000000    <1>       db     00h, 60h, 00h, 00h, 00h, 00h, 00h, 00h
00007220 598D5732005773E3    <1>       db     59h, 8Dh, 57h, 32h, 00h, 57h, 73h, 0E3h
00007228 FF                  <1>       db     0FFh   ; crtc regs
00007229 0001020304051407    <1>       db     00h, 01h, 02h, 03h, 04h, 05h, 14h, 07h
00007231 38393A3B3C3D3E3F    <1>       db     38h, 39h, 3Ah, 3Bh, 3Ch, 3Dh, 3Eh, 3Fh
00007239 01000F00            <1>       db     01h, 00h, 0Fh, 00h  ; actl regs
0000723D 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
00007246 2818080020          <1>       db     40, 24, 8, 00h, 20h ; tw, th-1, ch, slength
0000724B 090F0006            <1>       db     09h, 0Fh, 00h, 06h ; sequ regs
0000724F 63                  <1>       db     63h    ; misc reg
00007250 2D2728902B80BF1F    <1>       db     2Dh, 27h, 28h, 90h, 2Bh, 80h, 0BFh, 1Fh
00007258 00C0000000000000    <1>       db     00h, 0C0h, 00h, 00h, 00h, 00h, 00h, 00h
00007260 9C8E8F140096B9E3    <1>       db     9Ch, 8Eh, 8Fh, 14h, 00h, 96h, 0B9h, 0E3h
00007268 FF                  <1>       db     0FFh   ; crtc regs
00007269 0001020304050607    <1>       db     00h, 01h, 02h, 03h, 04h, 05h, 06h, 07h
00007271 1011121314151617    <1>       db     10h, 11h, 12h, 13h, 14h, 15h, 16h, 17h
00007279 01000F00            <1>       db     01h, 00h, 0Fh, 00h  ; actl regs
0000727D 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
00007286 5018080040          <1>       db     80, 24, 8, 00h, 40h ; tw, th-1, ch, slength
0000728B 010F0006            <1>       db     01h, 0Fh, 00h, 06h ; sequ regs
0000728F 63                  <1>       db     63h    ; misc reg
00007290 5F4F50825480BF1F    <1>       db     5Fh, 4Fh, 50h, 82h, 54h, 80h, 0BFh, 1Fh
00007298 00C0000000000000    <1>       db     00h, 0C0h, 00h, 00h, 00h, 00h, 00h, 00h
000072A0 9C8E8F280096B9E3    <1>       db     9Ch, 8Eh, 8Fh, 28h, 00h, 96h, 0B9h, 0E3h
000072A8 FF                  <1>       db     0FFh   ; crtc regs
000072A9 0001020304050607    <1>       db     00h, 01h, 02h, 03h, 04h, 05h, 06h, 07h
000072B1 1011121314151617    <1>       db     10h, 11h, 12h, 13h, 14h, 15h, 16h, 17h
000072B9 01000F00            <1>       db     01h, 00h, 0Fh, 00h  ; actl regs
000072BD 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
000072C6 50180E0080          <1>       db     80, 24, 14, 00h, 80h ; tw, th-1, ch, slength
000072CB 010F0006            <1>       db     01h, 0Fh, 00h, 06h ; sequ regs
000072CF A3                  <1>       db     0A3h   ; misc reg
000072D0 5F4F50825480BF1F    <1>       db     5Fh, 4Fh, 50h, 82h, 54h, 80h, 0BFh, 1Fh
000072D8 0040000000000000    <1>       db     00h, 40h, 00h, 00h, 00h, 00h, 00h, 00h
000072E0 83855D280F63BAE3    <1>       db     83h, 85h, 5Dh, 28h, 0Fh, 63h, 0BAh, 0E3h
000072E8 FF                  <1>       db     0FFh   ; crtc regs
000072E9 0001020304051407    <1>       db     00h, 01h, 02h, 03h, 04h, 05h, 14h, 07h
000072F1 38393A3B3C3D3E3F    <1>       db     38h, 39h, 3Ah, 3Bh, 3Ch, 3Dh, 3Eh, 3Fh
000072F9 01000F00            <1>       db     01h, 00h, 0Fh, 00h  ; actl regs
000072FD 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
00007306 501D1000A0          <1>       db     80, 29, 16, 00h, 0A0h ; tw, th-1, ch, slength
0000730B 010F0006            <1>       db     01h, 0Fh, 00h, 06h ; sequ regs
0000730F E3                  <1>       db     0E3h   ; misc reg
00007310 5F4F508254800B3E    <1>       db     5Fh, 4Fh, 50h, 82h, 54h, 80h, 0Bh, 3Eh
00007318 0040000000000000    <1>       db     00h, 40h, 00h, 00h, 00h, 00h, 00h, 00h
00007320 EA8CDF2800E704C3    <1>       db     0EAh, 8Ch, 0DFh, 28h, 00h, 0E7h, 04h, 0C3h ; 11/11/2020
00007328 FF                  <1>       db     0FFh   ; crtc regs
00007329 003F003F003F003F    <1>       db     00h, 3Fh, 00h, 3Fh, 00h, 3Fh, 00h, 3Fh
00007331 003F003F003F003F    <1>       db     00h, 3Fh, 00h, 3Fh, 00h, 3Fh, 00h, 3Fh
00007339 01000F00            <1>       db     01h, 00h, 0Fh, 00h  ; actl regs
0000733D 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 */
00007346 0001800290010800    <1>       dw 100h,  640, 400,  8
0000734E 01018002E0010800    <1>       dw 101h,  640, 480,  8
00007356 0301200358020800    <1>       dw 103h,  800, 600,  8
0000735E 0501000400030800    <1>       dw 105h, 1024, 768,  8
00007366 0E014001C8001000    <1>       dw 10Eh,  320, 200, 16
0000736E 0F014001C8001800    <1>       dw 10Fh,  320, 200, 24
00007376 11018002E0011000    <1>       dw 111h,  640, 480, 16
0000737E 12018002E0011800    <1>       dw 112h,  640, 480, 24
00007386 1401200358021000    <1>       dw 114h,  800, 600, 16
0000738E 1501200358021800    <1>       dw 115h,  800, 600, 24
00007396 1701000400031000    <1>       dw 117h, 1024, 768, 16
0000739E 1801000400031800    <1>       dw 118h, 1024, 768, 24
                             <1>
                             <1> ;/* BOCHS/PLEX86 'own' mode numbers */
000073A6 40014001C8002000    <1>       dw 140h,  320,  200, 32
000073AE 4101800290012000    <1>       dw 141h,  640,  400, 32
000073B6 42018002E0012000    <1>       dw 142h,  640,  480, 32
000073BE 4301200358022000    <1>       dw 143h,  800,  600, 32
000073C6 4401000400032000    <1>       dw 144h, 1024,  768, 32
000073CE 46014001C8000800    <1>       dw 146h,  320,  200,  8
000073D6 8D010005D0021000    <1>       dw 18Dh, 1280,  720, 16
000073DE 8E010005D0021800    <1>       dw 18Eh, 1280,  720, 24
000073E6 8F010005D0022000    <1>       dw 18Fh, 1280,  720, 32
000073EE 9001800738041000    <1>       dw 190h, 1920, 1080, 16
000073F6 9101800738041800    <1>       dw 191h, 1920, 1080, 24
000073FE 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
```

```
503                             <1> BnkNumberOfPages2:        db     0
504                             <1> LinNumberOfPages2:        db     0
505                             <1> LinRedMaskSize2:   db     0
506                             <1> LinRedFieldPosition2:     db     0
507                             <1> LinGreenMaskSize2:        db     0
508                             <1> LinGreenFieldPosition2: db       0
509                             <1> LinBlueMaskSize2:  db     0
510                             <1> LinBlueFieldPosition2:    db     0
511                             <1> LinRsvdMaskSize2:  db     0
512                             <1> LinRsvdFieldPosition2:    db     0
513                             <1> MaxPixelClock2:           dd     0
514                             <1>
515                             <1>                     dw     0103h ; 800x600x8
516                             <1> ModeAttributes3:    dw     MODE_ATTRIBUTES
517                             <1> WinAAttributes3:    db     WINA_ATTRIBUTES
518                             <1> WinBAttributes3:    db     0
519                             <1> WinGranularity3:    dw     VBE_DISPI_BANK_SIZE_KB
520                             <1> WinSize3:           dw     VBE_DISPI_BANK_SIZE_KB
521                             <1> WinASegment3:             dw     VGAMEM_GRAPH
522                             <1> WinBSegment3:             dw     0000h
523                             <1> WinFuncPtr3:        dd     0
524                             <1> BytesPerScanLine3: dw      800
525                             <1> XResolution3:             dw     800
526                             <1> YResolution3:             dw     600
527                             <1> XCharSize3:         db     8
528                             <1> YCharSize3:         db     16
529                             <1> NumberOfPlanes3:    db     1
530                             <1> BitsPerPixel3:            db     8
531                             <1> NumberOfBanks3:           db     8
532                             <1> MemoryModel3:             db     VBE_MEMORYMODEL_PACKED_PIXEL
533                             <1> BankSize3:          db     0
534                             <1> NumberOfImagePages3:      db     33
535                             <1> Reserved_page3:           db     0
536                             <1> RedMaskSize3:             db     0
537                             <1> RedFieldPosition3: db      0
538                             <1> GreenMaskSize3:           db     0
539                             <1> GreenFieldPosition3:      db     0
540                             <1> BlueMaskSize3:            db     0
541                             <1> BlueFieldPosition3:       db     0
542                             <1> RsvdMaskSize3:            db     0
543                             <1> RsvdFieldPosition3:       db     0
544                             <1> DirectColorModeInfo3:     db     0
545                             <1> PhysBasePtr3:             dd     VBE_DISPI_LFB_PHYSICAL_ADDRESS
546                             <1> OffScreenMemOffset3:      dd     0
547                             <1> OffScreenMemSize3: dw      0
548                             <1> LinBytesPerScanLine3:     dw     800
549                             <1> BnkNumberOfPages3:        db     0
550                             <1> LinNumberOfPages3:        db     0
551                             <1> LinRedMaskSize3:   db     0
552                             <1> LinRedFieldPosition3:     db     0
553                             <1> LinGreenMaskSize3:        db     0
554                             <1> LinGreenFieldPosition:    db     0
555                             <1> LinBlueMaskSize:   db     0
556                             <1> LinBlueFieldPosition:     db     0
557                             <1> LinRsvdMaskSize:   db     0
558                             <1> LinRsvdFieldPosition:     db     0
559                             <1> MaxPixelClock:            dd     0
560                             <1>
561                             <1>                     dw     0105h ; 1024x768x8
562                             <1> ModeAttributes4:    dw     MODE_ATTRIBUTES
563                             <1> WinAAttributes4:    db     WINA_ATTRIBUTES
564                             <1> WinBAttributes4:    db     0
565                             <1> WinGranularity4:    dw     VBE_DISPI_BANK_SIZE_KB
566                             <1> WinSize4:           dw     VBE_DISPI_BANK_SIZE_KB
567                             <1> WinASegment4:             dw     VGAMEM_GRAPH
568                             <1> WinBSegment4:             dw     0000h
569                             <1> WinFuncPtr4:        dd     0
570                             <1> BytesPerScanLine4: dw      1024
571                             <1> XResolution4:             dw     1024
572                             <1> YResolution4:             dw     768
573                             <1> XCharSize4:         db     8
574                             <1> YCharSize4:         db     16
575                             <1> NumberOfPlanes4:    db     1
576                             <1> BitsPerPixel4:            db     8
577                             <1> NumberOfBanks4:           db     12
578                             <1> MemoryModel4:             db     VBE_MEMORYMODEL_PACKED_PIXEL
579                             <1> BankSize4:          db     0
580                             <1> NumberOfImagePages4:      db     20
581                             <1> Reserved_page4:           db     0
582                             <1> RedMaskSize4:             db     0
583                             <1> RedFieldPosition4: db      0
584                             <1> GreenMaskSize4:           db     0
585                             <1> GreenFieldPosition4:      db     0
586                             <1> BlueMaskSize4:            db     0
587                             <1> BlueFieldPosition4:       db     0
588                             <1> RsvdMaskSize4:            db     0
589                             <1> RsvdFieldPosition4:       db     0
590                             <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
801                             <1> XResolution9:              dw     800
802                             <1> YResolution9:              dw     600
803                             <1> XCharSize9:        db     8
804                             <1> YCharSize9:        db     16
805                             <1> NumberOfPlanes9:   db     1
806                             <1> BitsPerPixel9:             db     16
807                             <1> NumberOfBanks9:            db     15
808                             <1> MemoryModel9:              db     VBE_MEMORYMODEL_DIRECT_COLOR
809                             <1> BankSize9:         db     0
810                             <1> NumberOfImagePages9:       db     16
811                             <1> Reserved_page9:            db     0
812                             <1> RedMaskSize9:              db     5
813                             <1> RedFieldPosition9: db     11
814                             <1> GreenMaskSize9:            db     6
815                             <1> GreenFieldPosition9:       db     5
816                             <1> BlueMaskSize9:             db     5
817                             <1> BlueFieldPosition9:        db     0
```

```
818                                     <1> RsvdMaskSize9:            db    0
819                                     <1> RsvdFieldPosition9:       db    0
820                                     <1> DirectColorModeInfo9:     db    0
821                                     <1> PhysBasePtr9:             dd    VBE_DISPI_LFB_PHYSICAL_ADDRESS
822                                     <1> OffScreenMemOffset9:      dd    0
823                                     <1> OffScreenMemSize9: dw     0
824                                     <1> LinBytesPerScanLine9:     dw    1600
825                                     <1> BnkNumberOfPages9:        db    0
826                                     <1> LinNumberOfPages9:        db    0
827                                     <1> LinRedMaskSize9:   db     5
828                                     <1> LinRedFieldPosition9:     db    11
829                                     <1> LinGreenMaskSize9: db     6
830                                     <1> LinGreenFieldPosition9:   db    5
831                                     <1> LinBlueMaskSize9:  db     5
832                                     <1> LinBlueFieldPosition9:    db    0
833                                     <1> LinRsvdMaskSize9:  db     0
834                                     <1> LinRsvdFieldPosition9:    db    0
835                                     <1> MaxPixelClock9:           dd    0
836                                     <1>
837                                     <1>                     dw    0115h ; 800x600x24
838                                     <1> ModeAttributes10:  dw     MODE_ATTRIBUTES
839                                     <1> WinAAttributes10:  db     WINA_ATTRIBUTES
840                                     <1> WinBAttributes10:  db     0
841                                     <1> WinGranularity10:  dw     VBE_DISPI_BANK_SIZE_KB
842                                     <1> WinSize10:         dw     VBE_DISPI_BANK_SIZE_KB
843                                     <1> WinASegment10:            dw    VGAMEM_GRAPH
844                                     <1> WinBSegment10:            dw    0000h
845                                     <1> WinFuncPtr10:             dd    0
846                                     <1> BytesPerScanLine10:       dw    2400
847                                     <1> XResolution10:            dw    800
848                                     <1> YResolution10:            dw    600
849                                     <1> XCharSize10:       db     8
850                                     <1> YCharSize10:       db     16
851                                     <1> NumberOfPlanes10:  db     1
852                                     <1> BitsPerPixel10:           db    24
853                                     <1> NumberOfBanks10:   db     22
854                                     <1> MemoryModel10:            db    VBE_MEMORYMODEL_DIRECT_COLOR
855                                     <1> BankSize10:        db     0
856                                     <1> NumberOfImagePages10:     db    10
857                                     <1> Reserved_page10:   db     0
858                                     <1> RedMaskSize10:            db    8
859                                     <1> RedFieldPosition10:       db    16
860                                     <1> GreenMaskSize10:   db     8
861                                     <1> GreenFieldPosition10:     db    8
862                                     <1> BlueMaskSize10:           db    8
863                                     <1> BlueFieldPosition10:      db    0
864                                     <1> RsvdMaskSize10:           db    0
865                                     <1> RsvdFieldPosition10:      db    0
866                                     <1> DirectColorModeInfo10:    db    0
867                                     <1> PhysBasePtr10:            dd    VBE_DISPI_LFB_PHYSICAL_ADDRESS
868                                     <1> OffScreenMemOffset10:     dd    0
869                                     <1> OffScreenMemSize10:       dw    0
870                                     <1> LinBytesPerScanLine10:    dw    2400
871                                     <1> BnkNumberOfPages10:       db    0
872                                     <1> LinNumberOfPages10:       db    0
873                                     <1> LinRedMaskSize10:  db     8
874                                     <1> LinRedFieldPosition10:    db    16
875                                     <1> LinGreenMaskSize10:       db    8
876                                     <1> LinGreenFieldPosition10:db      8
877                                     <1> LinBlueMaskSize10: db     8
878                                     <1> LinBlueFieldPosition10:   db    0
879                                     <1> LinRsvdMaskSize10: db     0
880                                     <1> LinRsvdFieldPosition10:   db    0
881                                     <1> MaxPixelClock10:   dd     0
882                                     <1>
883                                     <1>                     dw    0117h ; 1024x768x16
884                                     <1> ModeAttributes11:  dw     MODE_ATTRIBUTES
885                                     <1> WinAAttributes11:  db     WINA_ATTRIBUTES
886                                     <1> WinBAttributes11:  db     0
887                                     <1> WinGranularity11:  dw     VBE_DISPI_BANK_SIZE_KB
888                                     <1> WinSize11:         dw     VBE_DISPI_BANK_SIZE_KB
889                                     <1> WinASegment11:            dw    VGAMEM_GRAPH
890                                     <1> WinBSegment11:            dw    0000h
891                                     <1> WinFuncPtr11:             dd    0
892                                     <1> BytesPerScanLine11:       dw    2048
893                                     <1> XResolution11:            dw    1024
894                                     <1> YResolution11:            dw    768
895                                     <1> XCharSize11:       db     8
896                                     <1> YCharSize11:       db     16
897                                     <1> NumberOfPlanes11:  db     1
898                                     <1> BitsPerPixel11:           db    16
899                                     <1> NumberOfBanks11:   db     24
900                                     <1> MemoryModel11:            db    VBE_MEMORYMODEL_DIRECT_COLOR
901                                     <1> BankSize11:        db     0
902                                     <1> NumberOfImagePages11:     db    9
903                                     <1> Reserved_page11:   db     0
904                                     <1> RedMaskSize11:            db    5
905                                     <1> RedFieldPosition11:       db    11
906                                     <1> GreenMaskSize11:   db     6
907                                     <1> GreenFieldPosition11:     db    5
908                                     <1> BlueMaskSize11:           db    5
909                                     <1> BlueFieldPosition11:      db    0
910                                     <1> RsvdMaskSize11:           db    0
911                                     <1> RsvdFieldPosition11:      db    0
912                                     <1> DirectColorModeInfo11:    db    0
913                                     <1> PhysBasePtr11:            dd    VBE_DISPI_LFB_PHYSICAL_ADDRESS,
914                                     <1> OffScreenMemOffset11:     dd    0
915                                     <1> OffScreenMemSize11:       dw    0
916                                     <1> LinBytesPerScanLine11:    dw    2048
917                                     <1> BnkNumberOfPages11:       db    0
918                                     <1> LinNumberOfPages11:       db    0
919                                     <1> LinRedMaskSize11:  db     5
920                                     <1> LinRedFieldPosition11:    db    11
921                                     <1> LinGreenMaskSize11:       db    6
922                                     <1> LinGreenFieldPosition11:db      5
```

```
                        <1> LinBlueMaskSize11: db    5
                        <1> LinBlueFieldPosition11:  db    0
                        <1> LinRsvdMaskSize11: db    0
                        <1> LinRsvdFieldPosition11:  db    0
                        <1> MaxPixelClock11:   dd    0
                        <1>
                        <1>                    dw    0118h ; 1024x768x24
                        <1> ModeAttributes12:  dw    MODE_ATTRIBUTES
                        <1> WinAAttributes12:  db    WINA_ATTRIBUTES
                        <1> WinBAttributes12:  db    0
                        <1> WinGranularity12:  dw    VBE_DISPI_BANK_SIZE_KB
                        <1> WinSize12:         dw    VBE_DISPI_BANK_SIZE_KB
                        <1> WinASegment12:             dw    VGAMEM_GRAPH
                        <1> WinBSegment12:             dw    0000h
                        <1> WinFuncPtr12:              dd    0
                        <1> BytesPerScanLine12:        dw    3072
                        <1> XResolution12:             dw    1024
                        <1> YResolution12:             dw    768
                        <1> XCharSize12:       db    8
                        <1> YCharSize12:       db    16
                        <1> NumberOfPlanes12:  db    1
                        <1> BitsPerPixel12:            db    24
                        <1> NumberOfBanks12:   db    36
                        <1> MemoryModel12:             db    VBE_MEMORYMODEL_DIRECT_COLOR
                        <1> BankSize12:        db    0
                        <1> NumberOfImagePages12:      db    6
                        <1> Reserved_page12:   db    0
                        <1> RedMaskSize12:             db    8
                        <1> RedFieldPosition12:        db    16
                        <1> GreenMaskSize12:   db    8
                        <1> GreenFieldPosition12:      db    8
                        <1> BlueMaskSize12:            db    8
                        <1> BlueFieldPosition12:       db    0
                        <1> RsvdMaskSize12:            db    0
                        <1> RsvdFieldPosition12:       db    0
                        <1> DirectColorModeInfo12:     db    0
                        <1> PhysBasePtr12:             dd    VBE_DISPI_LFB_PHYSICAL_ADDRESS
                        <1> OffScreenMemOffset12:      dd    0
                        <1> OffScreenMemSize12:        dw    0
                        <1> LinBytesPerScanLine12:     dw    3072
                        <1> BnkNumberOfPages12:        db    0
                        <1> LinNumberOfPages12:        db    0
                        <1> LinRedMaskSize12:  db    8
                        <1> LinRedFieldPosition12:     db    16
                        <1> LinGreenMaskSize12:        db    8
                        <1> LinGreenFieldPosition12:db       8
                        <1> LinBlueMaskSize12: db    8
                        <1> LinBlueFieldPosition12:    db    0
                        <1> LinRsvdMaskSize12: db    0
                        <1> LinRsvdFieldPosition12:    db    0
                        <1> MaxPixelClock12:   dd    0
                        <1>
                        <1>                    dw    0140h ; 320x200x32
                        <1> ModeAttributes13:  dw    MODE_ATTRIBUTES
                        <1> WinAAttributes13:  db    WINA_ATTRIBUTES
                        <1> WinBAttributes13:  db    0
                        <1> WinGranularity13:  dw    VBE_DISPI_BANK_SIZE_KB
                        <1> WinSize13:         dw    VBE_DISPI_BANK_SIZE_KB
                        <1> WinASegment13:             dw    VGAMEM_GRAPH
                        <1> WinBSegment13:             dw    0000h
                        <1> WinFuncPtr13:              dd    0
                        <1> BytesPerScanLine13:        dw    1280
                        <1> XResolution13:             dw    320
                        <1> YResolution13:             dw    200
                        <1> XCharSize13:       db    8
                        <1> YCharSize13:       db    16
                        <1> NumberOfPlanes13:  db    1
                        <1> BitsPerPixel13:            db    32
                        <1> NumberOfBanks13:   db    4
                        <1> MemoryModel13:             db    VBE_MEMORYMODEL_DIRECT_COLOR
                        <1> BankSize13:        db    0
                        <1> NumberOfImagePages13:      db    64
                        <1> Reserved_page13:   db    0
                        <1> RedMaskSize13:             db    8
                        <1> RedFieldPosition13:        db    16
                        <1> GreenMaskSize13:   db    8
                        <1> GreenFieldPosition13:      db    8
                        <1> BlueMaskSize13:            db    8
                        <1> BlueFieldPosition13:       db    0
                        <1> RsvdMaskSize13:            db    8
                        <1> RsvdFieldPosition13:       db    24
                        <1> DirectColorModeInfo13:     db    VBE_DIRECTCOLOR_RESERVED_BITS_AVAILABLE
                        <1> PhysBasePtr13:             dd    VBE_DISPI_LFB_PHYSICAL_ADDRESS
                        <1> OffScreenMemOffset13:      dd    0
                        <1> OffScreenMemSize13:        dw    0
                        <1> LinBytesPerScanLine13:     dw    1280
                        <1> BnkNumberOfPages13:        db    0
                        <1> LinNumberOfPages13:        db    0
                        <1> LinRedMaskSize13:  db    8
                        <1> LinRedFieldPosition13:     db    16
                        <1> LinGreenMaskSize13:        db    8
                        <1> LinGreenFieldPosition13:db       8
                        <1> LinBlueMaskSize13: db    8
                        <1> LinBlueFieldPosition13:    db    0
                        <1> LinRsvdMaskSize13: db    8
                        <1> LinRsvdFieldPosition13:    db    24
                        <1> MaxPixelClock13:   dd    0
                        <1>
                        <1>                    dw    0141h ; 640x400x32
                        <1> ModeAttributes14:  dw    MODE_ATTRIBUTES
                        <1> WinAAttributes14:  db    WINA_ATTRIBUTES
                        <1> WinBAttributes14:  db    0
                        <1> WinGranularity14:  dw    VBE_DISPI_BANK_SIZE_KB
                        <1> WinSize14:         dw    VBE_DISPI_BANK_SIZE_KB
                        <1> WinASegment14:             dw    VGAMEM_GRAPH
```

```
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 00007406 00               <1> r_size_8:  db 0
1543 00007407 00               <1> r_pos_8:   db 0
1544 00007408 00               <1> g_size_8:  db 0
1545 00007409 00               <1> g_pos_8:   db 0
1546 0000740A 00               <1> b_size_8:  db 0
1547 0000740B 00               <1> b_pos_8:   db 0
1548 0000740C 00               <1> a_size_8:  db 0
1549 0000740D 00               <1> a_pos_8:   db 0
1550                            <1>
1551                            <1> ;case 16:
1552 0000740E 05               <1> r_size_16:  db 5
```

```
1553 0000740F 0B                <1> r_pos_16:   db 11
1554 00007410 06                <1> g_size_16:  db 6
1555 00007411 05                <1> g_pos_16:   db 5
1556 00007412 05                <1> b_size_16:  db 5
1557 00007413 00                <1> b_pos_16:   db 0
1558 00007414 00                <1> a_size_16:  db 0
1559 00007415 00                <1> a_pos_16:   db 0
1560                            <1>
1561                            <1> ;case 24:
1562 00007416 08                <1> r_size_24:  db 8
1563 00007417 10                <1> r_pos_24:   db 16
1564 00007418 08                <1> g_size_24:  db 8
1565 00007419 08                <1> g_pos_24:   db 8
1566 0000741A 08                <1> b_size_24:  db 8
1567 0000741B 00                <1> b_pos_24:   db 0
1568 0000741C 00                <1> a_size_24:  db 0
1569 0000741D 00                <1> a_pos_24:   db 0
1570                            <1>
1571                            <1> ;case 32:
1572 0000741E 08                <1> r_size_32:  db 8
1573 0000741F 10                <1> r_pos_32:   db 16
1574 00007420 08                <1> g_size_32:  db 8
1575 00007421 08                <1> g_pos_32:   db 8
1576 00007422 08                <1> b_size_32:  db 8
1577 00007423 00                <1> b_pos_32:   db 0
1578 00007424 08                <1> a_size_32:  db 8
1579 00007425 18                <1> a_pos_32:   db 24
3072                                ;%include 'diskdata.s'    ; DISK (BIOS) DATA (initialized)
3073                                ;;;
3074
3075                                Align 2
3076
3077                                %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)
  56                            <1> ; 19/04/2013 Retro UNIX 8086 v1 feaure only !
  57                            <1> ;;sdsegmnt equ     740h  ; swap data segment (for user structures and registers)
  58                            <1>
  59                            <1> ; 30/08/2013
  60                            <1> time_count equ 4 ; 10 --> 4 01/02/2014
  61                            <1>
  62                            <1> ; 05/02/2014
  63                            <1> ; process status
  64                            <1> ;SFREE       equ 0
  65                            <1> ;SRUN equ 1
  66                            <1> ;SWAIT       equ 2
  67                            <1> ;SZOMB       equ 3
  68                            <1> ;SSLEEP      equ 4 ; Retro UNIX 8086 V1 extension (for sleep and wakeup)
  69                            <1>
  70                            <1> ; 09/03/2015
  71                            <1> userdata equ 80000h ; user structure data address for current user ; temporary
  72                            <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 !'
```

```
178                                     <1> ; 16/06/2015
179                                     <1> ERR_NOT_DIR     equ 19 ; 'not a (valid) directory !' error
180                                     <1> ERR_FILE_SIZE         equ 20 ; 'file size error !'
181                                     <1> ; 22/06/2015
182                                     <1> ERR_NOT_SUPERUSER  equ 11 ; 'permission denied !' error
183                                     <1> ERR_NOT_OWNER      equ 11 ; 'permission denied !' error
184                                     <1> ERR_NOT_FILE       equ 11 ; 'permission denied !' error
185                                     <1> ; 23/06/2015
186                                     <1> ERR_FILE_EXISTS    equ 14 ; 'file already exists !' error
187                                     <1> ERR_DRV_NOT_SAME   equ 21 ; 'not same drive !' error
188                                     <1> ERR_DIR_NOT_FOUND  equ 12 ; 'directory not found !' error
189                                     <1> ERR_NOT_EXECUTABLE equ 22 ; 'not executable file !' error
190                                     <1> ; 27/06/2015
191                                     <1> ERR_INV_PARAMETER  equ 23 ; 'invalid parameter !' error
192                                     <1> ERR_INV_DEV_NAME   equ 24 ; 'invalid device name !' error
193                                     <1> ; 29/06/2015
194                                     <1> ERR_TIME_OUT    equ 25 ; 'time out !' error
195                                     <1> ERR_DEV_NOT_RESP   equ 25 ; 'device not responding !' error
196                                     <1> ; 10/10/2016
197                                     <1> ERR_INV_FILE_NAME  equ 26 ; 'invalid file name !' error
198                                     <1> ERR_INV_FLAGS         equ 23 ; 'invalid flags !' error
199                                     <1> ; For code compatibility with previous version of TRDOS (2011)
200                                     <1> ; (Temporary error codes for current TRDOS 386 -2016- version)
201                                     <1> ERR_NO_MORE_FILES  equ 12 ; 'no more files !' error
202                                     <1> ERR_PATH_NOT_FOUND equ  3 ; 'path not found !' error
203                                     <1>                       ; 'dir not found !' ; TRDOS 8086
204                                     <1> ERR_NOT_FOUND:        equ  2 ; 'file not found !' ; TRDOS 8086
205                                     <1> ERR_DISK_SPACE        equ 39 ; 'out of volume !' TRDOS 8086
206                                     <1>                      ; 'insufficient disk space !' ; 27h
207                                     <1> ERR_DISK_WRITE        equ 30 ; 'disk write protected !' ; 16/10/2016
208                                     <1> ERR_ACCESS_DENIED  equ  5 ; 'access denied !' ; TRDOS 8086
209                                     <1> ; 28/02/2017
210                                     <1> ERR_PERM_DENIED       equ 11 ; 'permission denied !' error
211                                     <1> ; 18/05/2016
212                                     <1> ERR_MISC        equ 27 ; miscellaneous/other errors
213                                     <1> ; 15/10/2016
214                                     <1> ; TRDOS 8086 -> TRDOS 386 (0Bh -> 28)
215                                     <1> ERR_INV_FORMAT        equ 28 ; 'invalid format !' error
216                                     <1> ; TRDOS 8086 -> TRDOS 386 (0Dh -> 29)
217                                     <1> ERR_INV_DATA    equ 29 ; 'invalid data !' error
218                                     <1> ; TRDOS 8086 -> TRDOS 386 (0Eh -> 20)
219                                     <1> ERR_ZERO_LENGTH       equ 20  ; 'zero length !' error
220                                     <1> ; TRDOS 8086 -> TRDOS 386 (15h -> 17, 1Dh -> 18, 1Eh -> 17)
221                                     <1> ERR_DRV_NR_READ       equ 17 ; 'drive not ready or read error !'
222                                     <1> ERR_DRV_NR_WRITE   equ 18 ; 'drive not ready or write error !'
223                                     <1> ; 15/10/2016
224                                     <1> ERR_INV_PATH_NAME  equ 19 ; 'bad path name !' error
225                                     <1> ERR_BAD_CMD_ARG       equ  1 ; 'bad command argument !' ; TRDOS 8086
226                                     <1> ERR_INV_FNUMBER       equ  1 ; 'invalid function number !' ; TRDOS 8086
227                                     <1> ERR_BIG_FILE    equ  8 ; 'big file & out of memory ! ; TRDOS 8086
228                                     <1> ERR_BIG_DATA    equ  8 ; 'big data & out of memory ! ; TRDOS 8086
229                                     <1> ERR_CLUSTER     equ 35 ; 'cluster not available !' ; TRDOS 8086
230                                     <1> ERR_OUT_OF_MEMORY  equ  4 ; 'out of memory !'
231                                     <1>                       ; 'insufficient memory !'
232                                     <1> ERR_P_VIOLATION       equ 6 ; 'protection violation !'
233                                     <1> ERR_PAGE_FAULT        equ 224 ;'page fault !' ;0E0h
234                                     <1> ERR_SWP_DISK_READ           equ 40
235                                     <1> ERR_SWP_DISK_NOT_PRESENT   equ 41
236                                     <1> ERR_SWP_SECTOR_NOT_PRESENT equ 42
237                                     <1> ERR_SWP_NO_FREE_SPACE      equ 43
238                                     <1> ERR_SWP_DISK_WRITE         equ 44
239                                     <1> ERR_SWP_NO_PAGE_TO_SWAP    equ 45
240                                     <1> ; 10/04/2017
241                                     <1> ERR_BUFFER     equ 46  ; 'buffer error !'
242                                     <1> ; 28/08/2017 (20/08/2017)
243                                     <1> ERR_DMA               equ -1  ; DMA buffer (allocation/misc.) error!
244                                     <1>
245                                     <1> ; 26/08/2015
246                                     <1> ; 24/07/2015
247                                     <1> ; 24/06/2015
248                                     <1> MAX_ARG_LEN     equ 256 ; max. length of sys exec arguments
249                                     <1> ; 01/07/2015
250                                     <1> MAX_MSG_LEN     equ 255 ; max. msg length for 'sysmsg'
251                                     <1> ;
252                                     <1> ; 06/10/2016
253                                     <1> OPENFILES       equ 10  ; max. number of open files (system)
254                                     <1> ; 07/10/2016
255                                     <1> ;NUMOFDEVICES         equ 20  ; max. num of available devices (sys)
256                                     <1>
3078                                        %include 'trdosk0.s' ; 04/01/2016
1                                       <1> ; ****************************************************************************
2                                       <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.0) - DEFINITIONS : trdosk0.s
3                                       <1> ; ----------------------------------------------------------------------------
4                                       <1> ; Last Update: 29/02/2016
5                                       <1> ; ----------------------------------------------------------------------------
6                                       <1> ; Beginning: 04/01/2016
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> ; TRDOS2.ASM (09/11/2011)
12                                      <1> ; ****************************************************************************
13                                      <1> ; TRDOS2.ASM (c) 2004-2011 Erdogan TAN [ 17/01/2004 ] Last Update: 09/11/2011
14                                      <1> ;
15                                      <1> ; Masterboot / Partition Table at Beginning+1BEh
16                                      <1> ptBootable       equ 0
17                                      <1> ptBeginHead      equ 1
18                                      <1> ptBeginSector    equ 2
19                                      <1> ptBeginCylinder  equ 3
20                                      <1> ptFileSystemID   equ 4
21                                      <1> ptEndHead        equ 5
22                                      <1> ptEndSector      equ 6
23                                      <1> ptEndCylinder    equ 7
24                                      <1> ptStartSector    equ 8
25                                      <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)
00007426 B036                   <1>       mov   al, 00110110b ; 36h
00007428 E643                   <1>       out   43h, al
0000742A 31C0                   <1>       xor   eax, eax  ; sub     al, al ; 0
0000742C E640                   <1>       out   40h, al ; LB
0000742E E640                   <1>       out   40h, al ; HB
                                <1>       ;
                                <1>       ; 30/03/2016
                                <1>       ; Clear Logical DOS Disk Description Tables Area
                                <1>       ;xor   eax, eax
00007430 BF00010900             <1>       mov   edi, Logical_DOSDisks
00007435 B980060000             <1>       mov   ecx, 6656/4 ; 26*256 = 6656 bytes
0000743A F3AB                   <1>       rep   stosd ; 1664 times 4 bytes
                                <1>
0000743C B83F3A2F00             <1>       mov   eax, '?:/'
00007441 A3[57820100]           <1>       mov   [Current_Dir_Drv], eax
                                <1>
                                <1>       ; Logical DRV INIT (only for hard disks)
00007446 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>
0000744B BE00010900             <1>       mov   esi, Logical_DOSDisks
00007450 B001                   <1>       mov   al, 1 ; Initialization sign (invalid_fd_parameter)
00007452 83C67E                 <1>       add   esi, LD_MediaChanged ; Media Change Status = 1 (init needed)
```

```
00007455 8806              <1>       mov    [esi], al ; A:
00007457 81C600010000      <1>       add    esi, 100h
0000745D 8806              <1>       mov    [esi], al ; B:
                           <1>
                           <1> _current_drive_bootdisk:
0000745F 8A15[EA6D0000]    <1>       mov    dl, [boot_drv] ; physical drive number
00007465 80FAFF            <1>       cmp    dl, 0FFh
00007468 740A              <1>       je     short _last_dos_diskno_check
                           <1> _boot_drive_check:
0000746A 80FA80            <1>       cmp    dl, 80h
0000746D 7218              <1>       jb     short _current_drive_a
0000746F 80EA7E            <1>       sub    dl, 7Eh ; C = 2 , D = 3
00007472 EB13              <1>       jmp    short _current_drive_a
                           <1>
                           <1> _last_dos_diskno_check:
00007474 8A15[22380100]    <1>       mov    dl, [Last_DOS_DiskNo]
0000747A 80FA02            <1>       cmp    dl, 2
0000747D 7706              <1>       ja     short _current_drive_c
0000747F 7406              <1>       je     short _current_drive_a
00007481 30D2              <1>       xor    dl, dl ; A:
00007483 EB02              <1>       jmp    short _current_drive_a
                           <1>
                           <1> _current_drive_c:
00007485 B202              <1>       mov    dl, 2 ; C:
                           <1>
                           <1> _current_drive_a:
00007487 8815[EB6D0000]    <1>       mov    [drv], dl
0000748D BE[24380100]      <1>         mov     esi, msg_CRLF_temp
00007492 E8AE000000        <1>       call   print_msg
                           <1>
00007497 8A15[EB6D0000]    <1>       mov    dl, [drv]
                           <1> _default_drive_c:
0000749D E82C0C0000        <1>       call   change_current_drive
000074A2 731C              <1>       jnc    short _start_mainprog
                           <1>
                           <1> _drv_not_ready_error:
000074A4 BE[DF3A0100]      <1>       mov    esi, msgl_drv_not_ready
000074A9 E897000000        <1>       call   print_msg
                           <1>         ;jmp _end_of_mainprog
                           <1>
                           <1>       ; 20/01/2018
000074AE B202              <1>       mov    dl, 2
000074B0 3815[EB6D0000]    <1>       cmp    [drv], dl
000074B6 736B              <1>       jnb    short _end_of_mainprog
000074B8 8815[EB6D0000]    <1>       mov    [drv], dl
000074BE 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
000074C0 66B80100          <1>       mov    ax, 1
000074C4 A2[B3030300]      <1>       mov    [u.uno], al
000074C9 66A3[4E030300]    <1>       mov    [mpid], ax
000074CF 66A3[20000300]    <1>       mov    [p.pid], ax
000074D5 A2[B0000300]      <1>       mov    [p.stat], al
000074DA C605[A8030300]04  <1>       mov    byte [u.quant], time_count  ; 07/01/2017
                           <1>       ;
000074E1 A1[90810100]      <1>       mov    eax, [k_page_dir]
000074E6 A3[B8030300]      <1>       mov    [u.pgdir], eax ; reset
                           <1>       ;
000074EB E8D9E6FFFF        <1>       call   allocate_page
000074F0 0F82B5000000      <1>       jc     panic
000074F6 A3[B4030300]      <1>       mov    [u.upage], eax ; user structure page
000074FB A3[C0000300]      <1>       mov    [p.upage], eax
00007500 E83EE7FFFF        <1>       call   clear_page
                           <1>       ;
                           <1>       ; 24/08/2015
00007505 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
0000750B BF00300900        <1>       mov    edi, Env_Page ; 93000h
00007510 B980000000        <1>       mov    ecx, Env_Page_Size / 4    ; 512/4  (4096/4)

00007515 31C0              <1>       xor    eax, eax
00007517 F3AB              <1>       rep    stosd
                           <1>
                           <1>       ; 14/04/2016
00007519 E807350000        <1>       call   mainprog_startup_configuration
                           <1>
0000751E E8EC0C0000        <1>         call    dos_prompt
                           <1>
                           <1> _end_of_mainprog:
00007523 BE[24380100]      <1>         mov     esi, msg_CRLF_temp
00007528 E818000000        <1>       call   print_msg
0000752D BE[2A380100]      <1>       mov    esi, mainprog_Version
00007532 E80E000000        <1>       call   print_msg
                           <1>       ; 24/01/2016
00007537 28E4              <1>       sub    ah, ah
00007539 E8A799FFFF        <1>       call   int16h ; call getch
0000753E E9879EFFFF        <1>       jmp    cpu_reset
                           <1>
00007543 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>       ;
00007545 8A3D[BE810100]     <1>       mov   bh, [ACTIVE_PAGE] ; 04/01/2016 (ptty)
                            <1>       ;mov  bl, 07h ; Black background, light gray forecolor
                            <1>
0000754B AC                 <1>       lodsb
                            <1> pmsg1:
0000754C 56                 <1>       push  esi
                            <1>       ;mov  bh, [ACTIVE_PAGE] ; 04/01/2016 (ptty)
0000754D B307               <1>       mov   bl, 07h ; Black background, light gray forecolor
0000754F E8A9ADFFFF         <1>       call  _write_tty
00007554 5E                 <1>       pop   esi
00007555 AC                 <1>       lodsb
00007556 20C0               <1>       and   al, al
00007558 75F2               <1>       jnz   short pmsg1
0000755A 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
0000755B 0FB61D[BE810100]   <1>       movzx ebx, byte [ACTIVE_PAGE] ; video page number (0 to 7)
00007562 8AA3[CB6F0000]     <1>       mov   ah, [ebx+vmode] ; default = 03h (80x25 text)
00007568 80FC04             <1>       cmp   ah, 4
0000756B 7205               <1>       jb    short cls1
0000756D 80FC07             <1>       cmp   ah, 7
00007570 7530               <1>       jne   short vga_clear
                            <1> cls1:
                            <1>       ;mov  bh, bl
                            <1>       ;mov  bl, 7
00007572 3A25[BA6F0000]     <1>       cmp   ah, [CRT_MODE] ; current video mode ?
00007578 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
0000757A 803D[1C120300]00   <1>       cmp   byte [pmi32], 0
00007581 771F               <1>       ja    short vga_clear
00007583 E97FADFFFF         <1>       jmp   set_mode_3
                            <1> cls2:
00007588 88DF               <1>       mov   bh, bl ; video page (0 to 7)
0000758A B307               <1>       mov   bl, 07h ; attribute to be used on blanked line
0000758C 28C0               <1>       sub   al, al ; 0 =  entire window
0000758E 6631C9             <1>       xor   cx, cx
00007591 66BA4F18           <1>       mov   dx, 184Fh
00007595 E8A3AAFFFF         <1>       call  _scroll_up ; 24/01/2016
                            <1>       ;
                            <1>       ;mov  bh, [ACTIVE_PAGE] ; video page number (0 to 7)
0000759A 6631D2             <1>       xor   dx, dx
                            <1>       ;call _set_cpos ; 24/01/2016
                            <1>       ;;retn
                            <1>       ; 03/12/2020
0000759D E9F5ADFFFF         <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)
000075A2 88E0               <1>       mov   al, ah
000075A4 28E4               <1>       sub   ah, ah  ; set current video mode
                            <1>       ;call _int10h ; simulates int 10h in TRDOS 386 kernel
                            <1>       ;jmp  short cls3
000075A6 E9BEA1FFFF         <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)
000075AB BE[E7440100]       <1>       mov   esi, panic_msg
000075B0 E890FFFFFF         <1>       call  print_msg
                            <1> key_to_reboot:
                            <1>         ; 24/01/2016
000075B5 28E4               <1>         sub     ah, ah
000075B7 E82999FFFF         <1>         call    int16h ; call   getch
                            <1>         ; wait for a character from the current tty
                            <1>       ;
000075BC B00A               <1>       mov   al, 0Ah
000075BE 8A3D[BE810100]     <1>       mov   bh, [ptty] ; [ACTIVE_PAGE]
000075C4 B307               <1>       mov   bl, 07h ; Black background,
                            <1>                     ; light gray forecolor
000075C6 E832ADFFFF         <1>       call  _write_tty
000075CB E9FA9DFFFF         <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>             ;
000075D0 66833D[AA030300]00 <1>       cmp   word [u.intr], 0
000075D8 7645               <1>       jna   short cbrk4
                            <1> cbrk0:
                            <1>       ; 12/11/2015
                            <1>       ; 06/12/2013
000075DA 66833D[AC030300]00 <1>       cmp   word [u.quit], 0
000075E2 743B               <1>       jz    short cbrk4
```

```
                            <1>       ;
                            <1>       ; 20/09/2013
000075E4 6650               <1>       push   ax
000075E6 A0[BE810100]       <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>       ;
000075EB 3C08               <1>       cmp    al, 8 ; serial port tty nums > 7
000075ED 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>       ;
000075EF 52                 <1>       push   edx
000075F0 0FB615[B3030300]   <1>       movzx  edx, byte [u.uno]
000075F7 3A82[7F000300]     <1>       cmp     al, [edx+p.ttyc-1] ; console tty (rw)
000075FD 5A                 <1>       pop    edx
000075FE 7412               <1>       je     short cbrk2
                            <1> cbrk1:
00007600 FEC0               <1>       inc    al  ; [u.ttyp] : 1 based tty number
                            <1>       ; 06/12/2013
00007602 3A05[94030300]     <1>       cmp    al, [u.ttyp] ; recent open tty (r)
00007608 7408               <1>       je     short cbrk2
0000760A 3A05[95030300]     <1>         cmp     al, [u.ttyp+1] ; recent open tty (w)
00007610 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>       ;
00007612 6631C0             <1>       xor    ax, ax ; 0
00007615 6648               <1>       dec    ax
                            <1>       ; 0FFFFh = 'ctrl+brk' keystroke
00007617 66A3[AC030300]     <1>       mov    [u.quit], ax
                            <1> cbrk3:
0000761D 6658               <1>       pop    ax
                            <1> cbrk4:
0000761F 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>
00007620 E87BF4FFFF         <1>       call   get_rtc_time        ; Return Current Time
00007625 86E9               <1>         xchg       ch,cl
00007627 66890D[5E7E0100]   <1>         mov  [hour], cx
0000762E 86F2               <1>         xchg       dh,dl
00007630 668915[627E0100]   <1>         mov  [second], dx
                            <1>       ;
00007637 E8D5F4FFFF         <1>         call       get_rtc_date        ; Return Current Date
0000763C 86E9               <1>         xchg       ch,cl
0000763E 66890D[587E0100]   <1>         mov  [year], cx
00007645 86F2               <1>         xchg       dh,dl
00007647 668915[5A7E0100]   <1>         mov  [month], dx
                            <1>       ;
0000764E 66B93030           <1>       mov    cx, 3030h
                            <1>       ;
00007652 A0[5E7E0100]       <1>       mov    al, [hour] ; Hour
                            <1>              ; AL <= BCD number)
00007657 D410               <1>         db   0D4h,10h            ; Undocumented inst. AAM
                            <1>                                  ; AH = AL / 10h
                            <1>                                  ; AL = AL MOD 10h
00007659 D50A               <1>         aad ; AX= AH*10+AL
0000765B A2[5E7E0100]       <1>       mov    [hour], al
00007660 A0[5F7E0100]       <1>       mov    al, [hour+1] ; Minute
                            <1>              ; AL <= BCD number)
00007665 D410               <1>         db   0D4h,10h            ; Undocumented inst. AAM
                            <1>                                  ; AH = AL / 10h
                            <1>                                  ; AL = AL MOD 10h
00007667 D50A               <1>         aad ; AX= AH*10+AL
```

```
00007669 A2[607E0100]        <1>       mov    [minute], al
0000766E A0[627E0100]        <1>       mov    al, [second] ; Second
                             <1>              ; AL <= BCD number)
00007673 D410                <1>         db   0D4h,10h            ; Undocumented inst. AAM
                             <1>                                  ; AH = AL / 10h
                             <1>                                  ; AL = AL MOD 10h
00007675 D50A                <1>         aad  ; AX= AH*10+AL
00007677 A2[627E0100]        <1>       mov    [second], al
0000767C 66A1[587E0100]      <1>       mov    ax, [year] ; Year (century)
00007682 6650                <1>         push       ax
                             <1>              ; AL <= BCD number)
00007684 D410                <1>         db   0D4h,10h            ; Undocumented inst. AAM
                             <1>                                  ; AH = AL / 10h
                             <1>                                  ; AL = AL MOD 10h
00007686 D50A                <1>         aad  ; AX= AH*10+AL
00007688 B464                <1>       mov    ah, 100
0000768A F6E4                <1>       mul    ah
0000768C 66A3[587E0100]      <1>       mov    [year], ax
00007692 6658                <1>       pop    ax
00007694 88E0                <1>       mov    al, ah
                             <1>              ; AL <= BCD number)
00007696 D410                <1>         db   0D4h,10h            ; Undocumented inst. AAM
                             <1>                                  ; AH = AL / 10h
                             <1>                                  ; AL = AL MOD 10h
00007698 D50A                <1>         aad  ; AX= AH*10+AL
0000769A 660105[587E0100]    <1>       add    [year], ax
000076A1 A0[5A7E0100]        <1>       mov    al, [month] ; Month
                             <1>              ; AL <= BCD number)
000076A6 D410                <1>         db   0D4h,10h            ; Undocumented inst. AAM
                             <1>                                  ; AH = AL / 10h
                             <1>                                  ; AL = AL MOD 10h
000076A8 D50A                <1>         aad  ; AX= AH*10+AL
000076AA A2[5A7E0100]        <1>       mov    [month], al
000076AF A0[5B7E0100]        <1>         mov     al, [month+1]          ; Day
                             <1>             ; AL <= BCD number)
000076B4 D410                <1>         db   0D4h,10h            ; Undocumented inst. AAM
                             <1>                                  ; AH = AL / 10h
                             <1>                                  ; AL = AL MOD 10h
000076B6 D50A                <1>         aad  ; AX= AH*10+AL
000076B8 A2[5C7E0100]        <1>         mov     [day], al
                             <1>
000076BD C3                  <1>       retn   ; 30/12/2017
                             <1>
                             <1> epoch:
000076BE 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
000076C3 29C0                <1>       sub    eax, eax
000076C5 66A1[587E0100]      <1>       mov    ax, [year]
000076CB 662DB207            <1>       sub    ax, 1970
000076CF BA6D010000          <1>       mov    edx, 365
000076D4 F7E2                <1>       mul    edx
000076D6 31DB                <1>       xor    ebx, ebx
000076D8 8A1D[5A7E0100]      <1>       mov    bl, [month]
000076DE FECB                <1>       dec    bl
000076E0 D0E3                <1>       shl    bl, 1
                             <1>       ;sub   edx, edx
000076E2 668B93[647E0100]    <1>       mov    dx, [EBX+DMonth]
000076E9 8A1D[5C7E0100]      <1>         mov     bl, [day]
000076EF FECB                <1>       dec    bl
000076F1 01D0                <1>       add    eax, edx
000076F3 01D8                <1>       add    eax, ebx
                             <1>                     ; EAX = days since 1/1/1970
000076F5 668B15[587E0100]    <1>       mov    dx, [year]
000076FC 6681EAB107          <1>       sub    dx, 1969
00007701 66D1EA              <1>       shr    dx, 1
00007704 66D1EA              <1>       shr    dx, 1
                             <1>              ; (year-1969)/4
00007707 01D0                <1>       add    eax, edx
                             <1>                     ; + leap days since 1/1/1970
00007709 803D[5A7E0100]02    <1>       cmp    byte [month], 2     ; if past february
00007710 7610                <1>       jna    short cte1
00007712 668B15[587E0100]    <1>       mov    dx, [year]
00007719 6683E203            <1>       and    dx, 3 ; year mod 4
0000771D 7503                <1>       jnz    short cte1
                             <1>                     ; and if leap year
0000771F 83C001              <1>       add    eax, 1       ; add this year's leap day (february 29)
                             <1> cte1:                     ; compute seconds since 1/1/1970
00007722 BA18000000          <1>       mov    edx, 24
00007727 F7E2                <1>       mul    edx
00007729 8A15[5E7E0100]      <1>       mov    dl, [hour]
0000772F 01D0                <1>       add    eax, edx
                             <1>              ; EAX = hours since 1/1/1970 00:00:00
                             <1>       ;mov   ebx, 60
00007731 B33C                <1>       mov    bl, 60
00007733 F7E3                <1>       mul    ebx
00007735 8A15[607E0100]      <1>       mov    dl, [minute]
0000773B 01D0                <1>       add    eax, edx
                             <1>              ; EAX = minutes since 1/1/1970 00:00:00
                             <1>       ;mov   ebx, 60
0000773D F7E3                <1>       mul    ebx
0000773F 8A15[627E0100]      <1>       mov    dl, [second]
00007745 01D0                <1>       add    eax, edx
                             <1>              ; EAX -> seconds since 1/1/1970 00:00:00
00007747 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>       ;
00007748 31D2                   <1>       xor    edx, edx
0000774A B93C000000             <1>       mov    ecx, 60
0000774F F7F1                   <1>       div    ecx
                                <1>       ;mov   [imin], eax   ; whole minutes
                                <1>                            ; since 1/1/1970
00007751 668915[627E0100]       <1>       mov    [second], dx  ; leftover seconds
00007758 29D2                   <1>       sub    edx, edx
0000775A F7F1                   <1>       div    ecx
                                <1>       ;mov   [ihrs], eax   ; whole hours
                                <1>       ;                   ; since 1/1/1970
0000775C 668915[607E0100]       <1>       mov    [minute], dx  ; leftover minutes
00007763 31D2                   <1>       xor    edx, edx
                                <1>       ;mov   cx, 24
00007765 B118                   <1>       mov    cl, 24
00007767 F7F1                   <1>       div    ecx
                                <1>       ;mov   [iday], ax   ; whole days
                                <1>                           ; since 1/1/1970
00007769 668915[5E7E0100]       <1>       mov    [hour], dx   ; leftover hours
00007770 05DB020000             <1>       add    eax, 365+366 ; whole day since
                                <1>                           ; 1/1/1968
                                <1>       ;mov   [iday], ax
00007775 50                     <1>       push   eax
00007776 29D2                   <1>       sub    edx, edx
00007778 B9B5050000             <1>       mov    ecx, (4*365)+1 ; 4 years = 1461 days
0000777D F7F1                   <1>       div    ecx
0000777F 59                     <1>       pop    ecx
                                <1>       ;mov   [lday], ax   ; count of quadyrs (4 years)
00007780 6652                   <1>       push   dx
                                <1>       ;mov   [qday], dx   ; days since quadyr began
00007782 6683FA3C               <1>       cmp    dx, 31 + 29  ; if past feb 29 then
00007786 F5                     <1>       cmc                 ; add this quadyr's leap day
00007787 83D000                 <1>       adc    eax, 0       ; to # of qadyrs (leap days)
                                <1>       ;mov   [lday], ax   ; since 1968
                                <1>       ;mov   cx, [iday]
0000778A 91                     <1>       xchg   ecx, eax     ; ECX = lday, EAX = iday
0000778B 29C8                   <1>       sub    eax, ecx     ; iday - lday
0000778D B96D010000             <1>       mov    ecx, 365
00007792 31D2                   <1>       xor    edx, edx
                                <1>       ; EAX = iday-lday, EDX = 0
00007794 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
00007796 6605B007               <1>       add    ax, 1968     ; compute year
0000779A 66A3[587E0100]         <1>       mov    [year], ax
000077A0 6689D1                 <1>       mov    cx, dx
                                <1>       ;mov   dx, [qday]
000077A3 665A                   <1>       pop    dx
000077A5 6681FA6D01             <1>       cmp    dx, 365           ; if qday <= 365 and qday >= 60
000077AA 7709                   <1>       ja     short cfe1   ; jday = jday +1
000077AC 6683FA3C               <1>       cmp    dx, 60       ; if past 2/29 and leap year then
000077B0 F5                     <1>         cmc                ; add a leap day to the # of whole
000077B1 6683D100               <1>       adc    cx, 0        ; days since 1/1 of current year
                                <1> cfe1:
                                <1>       ;mov   [jday], cx
000077B5 66BB0C00               <1>       mov    bx, 12       ; estimate month
000077B9 66BA6E01               <1>       mov    dx, 366      ; mday, max. days since 1/1 is 365
000077BD 6683E003               <1>       and    ax, 11b      ; year mod 4 (and dx, 3)
                                <1> cfe2: ; Month calculation  ; 0 to 11  (11 to 0)
000077C1 6639D1                 <1>       cmp    cx, dx       ; mday = # of days passed from 1/1
000077C4 731D                   <1>       jnb    short cfe3
000077C6 664B                   <1>       dec    bx           ; month = month - 1
000077C8 66D1E3                 <1>       shl    bx, 1
000077CB 668B93[647E0100]       <1>       mov    dx, [EBX+DMonth] ; # elapsed days at 1st of month
000077D2 66D1EB                 <1>       shr    bx, 1        ; bx = month - 1 (0 to 11)
000077D5 6683FB01               <1>       cmp    bx, 1        ; if month > 2 and year mod 4  = 0
000077D9 76E6                   <1>       jna    short cfe2   ; then mday = mday + 1
000077DB 08C0                   <1>       or     al, al       ; if past 2/29 and leap year then
000077DD 75E2                   <1>       jnz    short cfe2   ; add leap day (to mday)
000077DF 6642                   <1>       inc    dx           ; mday = mday + 1
000077E1 EBDE                   <1>       jmp    short cfe2
                                <1> cfe3:
000077E3 6643                   <1>       inc    bx          ; -> bx = month, 1 to 12
000077E5 66891D[5A7E0100]       <1>       mov    [month], bx
000077EC 6629D1                 <1>       sub    cx, dx      ; day = jday - mday + 1
000077EF 6641                   <1>       inc    cx
000077F1 66890D[5C7E0100]       <1>       mov    [day], cx
                                <1>
                                <1>       ; eax, ebx, ecx, edx is changed at return
                                <1>       ; output ->
                                <1>       ; [year], [month], [day], [hour], [minute], [second]
                                <1>
000077F8 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)
000077F9 E80F000000     <1>       call   set_date_bcd
                        <1>       ; Set real-time clock date
000077FE E83BF3FFFF     <1>       call   set_rtc_date ; RTC_50
                        <1>       ; Set real-time clock time
00007803 E832000000     <1>       call   set_time_bcd
00007808 E9C2F2FFFF     <1>       jmp    set_rtc_time ; RTC_30
                        <1>
                        <1> ; 31/12/2017
                        <1> set_date_bcd:
0000780D A0[597E0100]   <1>         mov     al, [year+1]
00007812 D40A           <1>       aam    ; ah = al / 10, al = al mod 10
00007814 D510           <1>       db     0D5h,10h     ; Undocumented inst. AAD
                        <1>                           ; AL = AH * 10h + AL
00007816 88C5           <1>       mov    ch, al ; century (BCD)
00007818 A0[587E0100]   <1>       mov    al, [year]
0000781D D40A           <1>       aam    ; ah = al / 10, al = al mod 10
0000781F D510           <1>       db     0D5h,10h     ; Undocumented inst. AAD
                        <1>                           ; AL = AH * 10h + AL
00007821 88C1           <1>       mov    cl, al ; year (BCD)
00007823 A0[5A7E0100]   <1>         mov  al, [month]
00007828 D40A           <1>       aam    ; ah = al / 10, al = al mod 10
0000782A D510           <1>       db     0D5h,10h     ; Undocumented inst. AAD
                        <1>                           ; AL = AH * 10h + AL
0000782C 88C6           <1>       mov    dh, al ; month (BCD)
0000782E A0[5C7E0100]   <1>       mov    al, [day]
00007833 D40A           <1>       aam    ; ah = al / 10, al = al mod 10
00007835 D510           <1>       db     0D5h,10h     ; Undocumented inst. AAD
                        <1>                           ; AL = AH * 10h + AL
00007837 88C6           <1>       mov    dh, al ; day (BCD)
00007839 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)
0000783A FA             <1>       cli
0000783B E850F4FFFF     <1>       CALL   UPD_IPR             ; CHECK FOR UPDATE IN PROCESS
                        <1>       ; cf = 1 -> al = 0
00007840 7207           <1>         jc      short stime1
00007842 B00B           <1>       MOV    AL,CMOS_REG_B       ; ADDRESS ALARM REGISTER
00007844 E862F4FFFF     <1>       CALL   CMOS_READ           ; READ CURRENT VALUE OF DSE BIT
                        <1> stime1:
00007849 FB             <1>       sti
0000784A 2401           <1>       AND    AL,00000001B        ; MASK FOR VALID DSE BIT
0000784C 88C2           <1>       MOV    DL,AL               ; SET [DL] TO ZERO FOR NO DSE BIT
                        <1>       ; DL = 1 or 0 (day light saving time)
                        <1>       ;
0000784E A0[5E7E0100]   <1>       mov    al, [hour]
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 88C5           <1>       mov    ch, al ; hour (BCD)
00007859 A0[607E0100]   <1>         mov     al, [minute]
0000785E D40A           <1>       aam    ; ah = al / 10, al = al mod 10
00007860 D510           <1>       db     0D5h,10h     ; Undocumented inst. AAD
                        <1>                           ; AL = AH * 10h + AL
00007862 88C1           <1>       mov    cl, al       ; minute (BCD)
00007864 A0[627E0100]   <1>         mov     al, [second]
00007869 D40A           <1>       aam    ; ah = al / 10, al = al mod 10
0000786B D510           <1>       db     0D5h,10h     ; Undocumented inst. AAD
                        <1>                           ; AL = AH * 10h + AL
0000786D 88C6           <1>       mov    dh, al      ; second (BCD)
0000786F 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>
00007870 A0[2C820100]   <1>       mov    al, [HF_NUM] ; number of fixed disks
00007875 20C0           <1>       and    al, al
```

```
00007877 7501                <1>       jnz    short load_hd_partition_tables
                             <1>
                             <1>       ; no any hard disks
00007879 C3                  <1>       retn
                             <1>
                             <1> load_hd_partition_tables:
                             <1>       ;mov   esi, [HDPM_TBL_VEC] ; primary master disk FDPT
                             <1>       ; 15/07/2020
0000787A BE[30820100]        <1>       mov    esi, HDPM_TBL_VEC
0000787F BF[56860100]        <1>       mov    edi, PTable_hd0
00007884 B280                <1>       mov    dl, 80h
                             <1>       ; 15/07/2020
00007886 A2[ED6D0000]        <1>       mov    [hdc], al
                             <1>       ;xor   ecx, ecx ; 0
                             <1> load_next_hd_partition_table:
                             <1>       ; 20/07/2020
0000788B 31C9                <1>       xor    ecx, ecx ; 0
                             <1>       ;push  ecx
0000788D 57                  <1>       push   edi ; *
                             <1>       ;push  esi ; FDPT (+ DPTE) address
                             <1>       ; 15/07/2020
0000788E AD                  <1>       lodsd
0000788F 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
00007890 8A4814              <1>       mov    cl, [eax+20]
00007893 80E140              <1>       and    cl, 40h
00007896 880D[5A870100]      <1>       mov    [HD_LBA_yes], cl
                             <1>
0000789C E844040000          <1>       call   load_masterboot
                             <1>       ;jc    short pass_pt_this_hard_disk
                             <1>       ; 13/08/2020
000078A1 0F828A000000        <1>       jc     pass_pt_this_hard_disk
                             <1>
000078A7 BB[14860100]        <1>       mov    ebx, PartitionTable
000078AC 89DE                <1>       mov    esi, ebx
                             <1>       ;mov   ecx, 16
000078AE B110                <1>       mov    cl, 16
000078B0 F3A5                <1>       rep    movsd
000078B2 89DE                <1>       mov    esi, ebx
                             <1>       ;mov   byte [hdc], 4 ; 4 - partition index
                             <1>       ; 15/07/2020
000078B4 C605[5B870100]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
000078BB 8A4604              <1>       mov    al, [esi+ptFileSystemID]
000078BE 20C0                <1>       and    al, al
000078C0 7457                <1>       jz     short loc_validate_next_hdp_partition2
                             <1>
000078C2 56                  <1>       push   esi ; *** ; Masterboot partition table offset
000078C3 52                  <1>       push   edx ; **** ; dl = Physical drive number
                             <1>
                             <1>       ; 13/08/2020
000078C4 3C05                <1>       cmp    al, 05h  ; Extended partition CHS
000078C6 7404                <1>       je     short loc_set_ep_counter
000078C8 3C0F                <1>       cmp    al, 0Fh  ; Extended partition LBA
000078CA 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
000078CC 803D[5C870100]80    <1>       cmp    byte [EP_Counter], 80h
000078D3 7342                <1>       jnb    short loc_validate_next_hdp_partition1
                             <1>
000078D5 8815[5C870100]      <1>       mov    byte [EP_Counter], dl ; disk drv has extd. part.
                             <1>
000078DB EB3A                <1>       jmp    short loc_validate_next_hdp_partition1
                             <1>
                             <1> loc_validate_next_hdp_partition0:
000078DD 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
000078DF E885010000          <1>       call   validate_hd_fat_partition
000078E4 730E                <1>       jnc    short loc_set_valid_hdp_partition_entry
                             <1>
                             <1>       ;pop   edx
                             <1>       ;push  edx
000078E6 8B1424              <1>       mov    edx, [esp]  ; ****
000078E9 8B742404            <1>       mov    esi, [esp+4] ; *** ; 30/01/2018
000078ED E8D1020000          <1>       call   validate_hd_fs_partition
000078F2 7223                <1>       jc     short loc_validate_next_hdp_partition1
                             <1> loc_set_valid_hdp_partition_entry:
000078F4 8A0D[22380100]      <1>       mov    cl, [Last_DOS_DiskNo]
000078FA 80C141              <1>       add    cl, 'A'
                             <1>       ; ESI = Logical dos drive description table address
000078FD 880E                <1>       mov    [esi+LD_Name], cl
                             <1>       ; 15/07/2020
000078FF 8A4602              <1>       mov    al, [esi+LD_PhyDrvNo] ; Physical drive number
                             <1>       ;mov   al, [esp] ; ****
00007902 2C7F                <1>       sub    al, 7Fh
                             <1>              ; AL = 1 to 4
00007904 C0E002              <1>       shl    al, 2 ; AL = 4 to 16
                             <1>
```

```
00007907 8A15[5B870100]      <1>       mov   dl, [PP_Counter]
                             <1>
                             <1>       ;sub   al, [PP_Counter]
0000790D 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
0000790F B404                <1>       mov   ah, 4
                             <1>       ;sub   ah, [PP_Counter]
00007911 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
00007913 6689467C            <1>       mov   [esi+LD_PartitionEntry], ax
                             <1>
                             <1> loc_validate_next_hdp_partition1:
00007917 5A                  <1>       pop   edx ; **** ; dl = Physical drive number
00007918 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
00007919 FE0D[5B870100]      <1>       dec   byte [PP_Counter] ; 4 - partition index
0000791F 7410                <1>       jz    short pass_pt_this_hard_disk
                             <1>
00007921 83C610              <1>       add   esi, 16 ; 10h
00007924 EB95                <1>       jmp   short loc_validate_hdp_partition
                             <1>
                             <1> loc_not_any_extd_partitions:
                             <1>       ; 15/07/2020
00007926 C3                  <1>       retn
                             <1>
                             <1> loc_next_hd_partition_table:
00007927 FEC2                <1>       inc   dl
                             <1>       ; 15/07/2020
                             <1>       ;add   esi, 32 ; next FDPT address
00007929 83C740              <1>       add   edi, 64 ; next partition table destination
0000792C 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
00007931 5E                  <1>       pop   esi ; ** ; next FDPT (+ DPTE) address ptr
00007932 5F                  <1>       pop   edi ; * ; Ptable_hd?
                             <1>       ;pop   ecx
                             <1>       ;loop  loc_next_hd_partition_table
00007933 FE0D[ED6D0000]      <1>       dec   byte [hdc]
00007939 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
0000793B A0[5C870100]        <1>       mov   al, [EP_Counter] ; 1st disk drv has extd partition
00007940 08C0                <1>       or    al, al ; 0 ?
00007942 74E2                <1>       jz    short loc_not_any_extd_partitions
                             <1>
                             <1> load_extended_dos_partitions:
                             <1>       ;mov   byte [hdc], 80h
                             <1>       ; 13/08/2020
00007944 A2[ED6D0000]        <1>       mov   byte [hdc], al ; 1st disk drv has extd partition
                             <1>       ; 25/08/2020
00007949 2C80                <1>       sub   al, 80h
0000794B 740E                <1>       jz    short loc_set_ext_ptable_hd0
0000794D C0E006              <1>       shl   al, 6 ; * 64
00007950 0FB6F0              <1>       movzx esi, al
00007953 81C6[56860100]      <1>       add   esi, PTable_hd0
00007959 EB05                <1>       jmp   short next_hd_extd_partition
                             <1>
                             <1>       ; 25/08/2020
                             <1> loc_set_ext_ptable_hd0:
0000795B BE[56860100]        <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
00007960 66C705[5C870100]00- <1>       mov   word [EP_Counter], 0 ; Reset EP index and LD index
00007968 00                  <1>
                             <1>
00007969 56                  <1>       push  esi ; **** ; PTable_hd? offset
                             <1>
0000796A C605[5B870100]04    <1>       mov   byte [PP_Counter], 4
                             <1>                          ; set for each extd partition table
                             <1>       ;;mov  ecx, 4
```

```
                            <1>       ;mov   cl, 4
00007971 8A15[ED6D0000]     <1>       mov    dl, [hdc]
                            <1> hd_check_fs_id_05h:
00007977 8A4604             <1>       mov    al, [esi+ptFileSystemID]
0000797A 3C05               <1>       cmp    al, 05h ; Is it an extended dos partition ?
0000797C 7411               <1>       je     short loc_set_ep_start_sector ; yes
                            <1> hd_check_fs_id_0Fh:
0000797E 3C0F               <1>       cmp    al, 0Fh ; Is it an extended win4 (LBA mode) partition ?
00007980 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
00007982 FE0D[5B870100]     <1>       dec    byte [PP_Counter] ; 4 --> 0
00007988 742D               <1>       jz     short continue_check_ep_next_disk
0000798A 83C610             <1>       add    esi, 16
0000798D 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
0000798F 8B4E08             <1>       mov    ecx, [esi+ptStartSector]
                            <1>       ; 30/08/2020
00007992 890D[5E870100]     <1>       mov    [MBR_EP_StartSector], ecx
                            <1>       ; 20/07/2020
                            <1> loc_validate_hde_partition_next:
00007998 890D[62870100]     <1>       mov    [EP_StartSector], ecx ; Extended partition's start sector
0000799E BB[56840100]       <1>         mov  ebx, MasterBootBuff
000079A3 803D[5A870100]01   <1>       cmp    byte [HD_LBA_yes], 1 ; LBA ready = Yes
000079AA 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
000079AC 66B8011B           <1>       mov    ax, 1B01h
000079B0 E86AD8FFFF         <1>       call   int13h
                            <1>       ;pop   ecx
                            <1>       ;jnc   short loc_hd_move_ep_table
                            <1>       ; 15/07/2020
000079B5 732E               <1>       jnc    short loc_validate_hde_partition
                            <1>
                            <1> continue_check_ep_next_disk:
                            <1>       ; 15/07/2020
                            <1>       ;pop   edi ; PTable_ep?
000079B7 5E                 <1>       pop    esi ; **** ; PTable_hd?
000079B8 A0[2C820100]       <1>       mov    al, [HF_NUM] ; number of hard disks
000079BD 047F               <1>       add    al, 7Fh
000079BF 3805[ED6D0000]     <1>       cmp    [hdc], al
000079C5 730B               <1>       jnb    short loc_validating_hd_partitions_ok
000079C7 83C640             <1>       add    esi, 64
                            <1>       ; 15/07/2020
                            <1>       ;add   edi, 64
000079CA FE05[ED6D0000]     <1>       inc    byte [hdc]
000079D0 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:
000079D2 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
000079D3 8A7601             <1>       mov    dh, [esi+ptBeginHead]
000079D6 668B4E02           <1>         mov     cx, [esi+ptBeginSector]
000079DA 66B80102           <1>       mov    ax, 0201h ; Read 1 sector
                            <1>       ;mov   ebx, MasterBootBuff
000079DE E83CD8FFFF         <1>       call   int13h ; 20/07/2020
                            <1>                     ; 'diskio.s' modification, 'clc'
                            <1>       ;pop   ecx
000079E3 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
000079E5 BE[14860100]       <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
000079EA 807E0400           <1>         cmp    byte [esi+ptFileSystemID], 0
000079EE 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
000079F0 FE05[5C870100]     <1>         inc    byte [EP_Counter] ; current (sub partition) index
                            <1>                               ; in current extended partition
                            <1>
000079F6 BF[62870100]       <1>         mov    edi, EP_StartSector
                            <1>
                            <1>         ; Input -> ESI = PartitionTable offset
                            <1>         ; DL = Hard disk drive number
                            <1>         ; EDI = Extended partition start sector pointer
000079FB E869000000         <1>         call   validate_hd_fat_partition
                            <1>         ;pop   ecx ; *
00007A00 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
00007A02 8A15[ED6D0000]     <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
00007A08 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
00007A0A A0[ED6D0000]       <1>         mov    al, [hdc] ; Hard disk drive number (>=80h)
00007A0F 88C2               <1>         mov    dl, al ; mov dl, [hdc]
00007A11 2C7F               <1>         sub    al, 7Fh
                            <1>                    ; 1 to 4
00007A13 C0E002             <1>         shl    al, 2 ; 4 to 16
00007A16 2A05[5B870100]     <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
00007A1C 8A25[5D870100]         <1>       mov   ah, [LD_Counter] ; Logical drive index number
                                <1>                              ; (in current extended partition)
00007A22 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
00007A25 6689467C               <1>       mov   [esi+LD_PartitionEntry], ax
                                <1>
00007A29 8A0D[22380100]         <1>       mov   cl, [Last_DOS_DiskNo]
00007A2F 80C141                 <1>       add   cl, 'A'
00007A32 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
00007A34 FE05[5D870100]         <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
00007A3A 803D[5D870100]04       <1>       cmp   byte [LD_Counter], 4 ; maximum 4 logical disks
                                <1>                                  ; per extended partition
00007A41 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
00007A47 BE[24860100]           <1>       mov   esi, PartitionTable+16
                                <1>
                                <1>       ; 20/07/2020
00007A4C 8A4604                 <1>       mov   al, [esi+ptFileSystemID]
                                <1>
                                <1>       ; 20/07/2020
00007A4F 3C05                   <1>       cmp   al, 05h ; Is it an extended dos partition ?
00007A51 7408                   <1>       je    short loc_minidisk_next_ep_lba_chs ; 17/07/2020
00007A53 3C0F                   <1>       cmp   al, 0Fh ; Is it an extended win4 (LBA mode) partition ?
00007A55 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
00007A5B 8B4E08                 <1>       mov   ecx, [esi+ptStartSector] ; relative start sector number
                                <1>       ;add  ecx, [EP_StartSector]
                                <1>       ; 30/08/2020
00007A5E 030D[5E870100]         <1>       add   ecx, [MBR_EP_StartSector]
                                <1>       ; 20/07/2020
00007A64 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
00007A69 8A6604         <1>       mov    ah, [esi+ptFileSystemID]
00007A6C B002           <1>       mov    al, 2 ; 27/12/2017
00007A6E 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)
00007A71 7711           <1>       ja     short vhdp_FAT16_32
00007A73 7425           <1>       je     short loc_set_valid_hd_partition_params
                        <1>
                        <1> vhdp_FAT12_16:
                        <1>       ; 27/12/2017
00007A75 FEC8           <1>       dec    al ; mov al, 1
00007A77 38C4           <1>       cmp    ah, al ; 1 ; FAT12 partition
00007A79 741F           <1>       je     short loc_set_valid_hd_partition_params
                        <1>       ;
00007A7B FEC0           <1>       inc    al ; mov al, 2
00007A7D 80FC04         <1>       cmp    ah, 04h ; FAT16 CHS partition (< 32MB)
00007A80 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:
00007A82 F9             <1>       stc
                        <1>       ; cf=1
00007A83 C3             <1>       retn
                        <1>
                        <1> vhdp_FAT16_32:
                        <1>       ; 15/07/2020
                        <1>       ;mov   al, 3
00007A84 FEC0           <1>       inc    al
00007A86 80FC0C         <1>       cmp    ah, 0Ch ; FAT32 LBA partition
00007A89 740F           <1>       je     short loc_set_valid_hd_partition_params
00007A8B 7706           <1>       ja     short vhdp_check_FAT16_lba
                        <1>
                        <1> vhdp_check_FAT32_chs:
00007A8D 80FC0B         <1>       cmp    ah, 0Bh ; FAT32 CHS partition
00007A90 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:
00007A92 C3             <1>       retn
                        <1>
                        <1> vhdp_check_FAT16_lba:
00007A93 80FC0E         <1>       cmp    ah, 0Eh ; FAT16 LBA partition
00007A96 75EA           <1>       jne    short loc_not_a_valid_fat_partition1
                        <1>
                        <1>       ;mov   al, 2
00007A98 FEC8           <1>       dec    al
                        <1>
                        <1> loc_set_valid_hd_partition_params:
                        <1>       ; 15/07/2020
                        <1>       ;inc   byte [Last_DOS_DiskNo] ; > 1
                        <1>       ;
00007A9A 31DB           <1>       xor    ebx, ebx
00007A9C 8A3D[22380100] <1>       mov    bh, [Last_DOS_DiskNo] ; * 256
00007AA2 FEC7           <1>       inc    bh ; 15/07/2020
00007AA4 81C300010900   <1>       add    ebx, Logical_DOSDisks
                        <1>       ;
00007AAA C6430102       <1>       mov    byte [ebx+LD_DiskType], 2
00007AAE 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
00007AB1 66894303       <1>       mov    word [ebx+LD_FATType], ax
                        <1>       ;
00007AB5 8B4E08         <1>       mov    ecx, [esi+ptStartSector]
00007AB8 09FF           <1>       or     edi, edi
00007ABA 7402           <1>       jz     short pass_hd_FAT_ep_start_sector_adding
                        <1> loc_add_hd_FAT_ep_start_sector:
                        <1>       ; 17/07/2020
00007ABC 030F           <1>       add    ecx, [edi]
                        <1> pass_hd_FAT_ep_start_sector_adding:
00007ABE 894B6C         <1>       mov    [ebx+LD_StartSector], ecx
                        <1> loc_hd_FAT_logical_drv_init:
00007AC1 89DD           <1>       mov    ebp, ebx
                        <1>       ;mov   dl, [ebx+LD_PhyDrvNo]
00007AC3 A0[5A870100]   <1>       mov    al, [HD_LBA_yes] ; 07/01/2016
00007AC8 884305         <1>       mov    [ebx+LD_LBAYes], al
00007ACB BB[66870100]   <1>       mov    ebx, DOSBootSectorBuff ; buffer address
00007AD0 08C0           <1>       or     al, al
00007AD2 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)
00007AD4 B41B           <1>       mov    ah, 1Bh ; LBA read
00007AD6 B001           <1>       mov    al, 1 ; sector count
00007AD8 E842D7FFFF     <1>       call   int13h
00007ADD 7313           <1>       jnc    short loc_hd_drv_FAT_boot_validation
                        <1> loc_not_a_valid_fat_partition3:
00007ADF C3             <1>       retn
                        <1> loc_hd_FAT_drv_init_load_bs_chs:
00007AE0 8A7601         <1>       mov    dh, [esi+ptBeginHead]
00007AE3 668B4E02       <1>       mov    cx, [esi+ptBeginSector]
00007AE7 66B80102       <1>       mov    ax, 0201h ; Read 1 sector
                        <1>       ;mov   ebx, DOSBootSectorBuff
```

```
00007AEB E82FD7FFFF          <1>       call   int13h
00007AF0 72ED                <1>       jc     short loc_not_a_valid_fat_partition3
                             <1> loc_hd_drv_FAT_boot_validation:
                             <1>       ;mov   esi, DOSBootSectorBuff
00007AF2 89DE                <1>       mov    esi, ebx
00007AF4 6681BEFE01000055AA  <1>       cmp    word [esi+BS_Validation], 0AA55h
00007AFD 7512                <1>       jne    short loc_not_a_valid_fat_partition4
00007AFF 807E15F8            <1>       cmp    byte [esi+BPB_Media], 0F8h
00007B03 750C                <1>       jne    short loc_not_a_valid_fat_partition4
                             <1>
                             <1>       ; 27/12/2017
00007B05 807D0303            <1>       cmp    byte [ebp+LD_FATType], 3
00007B09 7508                <1>       jne    short loc_hd_FAT16_BPB
                             <1>
                             <1> loc_hd_drv_FAT32_boot_validation:
00007B0B 807E4229            <1>       cmp    byte [esi+BS_FAT32_BootSig], 29h
00007B0F 7416                <1>       je     short loc_hd_FAT32_BPB
                             <1>
                             <1> loc_not_a_valid_fat_partition4:
00007B11 F9                  <1>       stc
00007B12 C3                  <1>       retn
                             <1>
                             <1> loc_hd_FAT16_BPB:
00007B13 807E2629            <1>       cmp    byte [esi+BS_BootSig], 29h
00007B17 75F8                <1>       jne    short loc_not_a_valid_fat_partition4
                             <1>
00007B19 66837E1600          <1>       cmp    word [esi+BPB_FATSz16], 0
00007B1E 7607                <1>       jna    short loc_hd_big_FAT16_BPB
00007B20 B920000000          <1>       mov    ecx, 32
00007B25 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:
00007B27 B92D000000          <1>       mov    ecx, 45
                             <1>
                             <1> loc_hd_move_FAT_BPB:
00007B2C 89EF                <1>       mov    edi, ebp
                             <1>       ;mov   esi, ebx ; Boot sector
00007B2E 57                  <1>       push   edi
00007B2F 83C706              <1>       add    edi, LD_BPB
00007B32 F366A5              <1>       rep    movsw
00007B35 5E                  <1>       pop    esi
00007B36 0FB74614            <1>       movzx  eax, word [esi+LD_BPB+BPB_RsvdSecCnt]
00007B3A 03466C              <1>       add    eax, [esi+LD_StartSector]
00007B3D 894660              <1>       mov    [esi+LD_FATBegin], eax
00007B40 807E0303            <1>       cmp    byte [esi+LD_FATType], 3
00007B44 7224                <1>       jb     short loc_set_FAT16_RootDirLoc
                             <1> loc_set_FAT32_RootDirLoc:
00007B46 8B462A              <1>       mov    eax, [esi+LD_BPB+BPB_FATSz32]
00007B49 0FB65E16            <1>         movzx      ebx, byte [esi+LD_BPB+BPB_NumFATs]
00007B4D F7E3                <1>       mul    ebx
00007B4F 034660              <1>       add    eax, [esi+LD_FATBegin]
                             <1> loc_set_FAT32_data_begin:
00007B52 894668              <1>       mov    [esi+LD_DATABegin], eax
00007B55 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.
00007B58 8B4632              <1>       mov    eax, [esi+LD_BPB+BPB_RootClus]
00007B5B 83E802              <1>       sub    eax, 2
00007B5E 7435                <1>       jz     short short loc_set_32bit_FAT_total_sectors
                             <1>       ;movzx ebx, byte [esi+LD_BPB+BPB_SecPerClust]
00007B60 8A5E13              <1>       mov    bl, [esi+LD_BPB+BPB_SecPerClust]
00007B63 F7E3                <1>       mul    ebx
00007B65 014664              <1>       add    [esi+LD_ROOTBegin], eax
00007B68 EB2B                <1>       jmp    short loc_set_32bit_FAT_total_sectors
                             <1>       ;
                             <1> loc_set_FAT16_RootDirLoc:
00007B6A 0FB64616            <1>       movzx  eax, byte [esi+LD_BPB+BPB_NumFATs]
00007B6E 0FB7561C            <1>       movzx  edx, word [esi+LD_BPB+BPB_FATSz16]
00007B72 F7E2                <1>       mul    edx
00007B74 034660              <1>       add    eax, [esi+LD_FATBegin]
00007B77 894664              <1>       mov    [esi+LD_ROOTBegin], eax
                             <1> loc_set_FAT16_data_begin:
00007B7A 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
00007B7D 0FB74617            <1>       movzx  eax, word [esi+LD_BPB+BPB_RootEntCnt]
00007B81 6683C00F            <1>       add    ax, 15
00007B85 66C1E804            <1>       shr    ax, 4 ; / 16 ; (16 entries per sector)
00007B89 014668              <1>       add    [esi+LD_DATABegin], eax
                             <1>       ;movzx eax, word [esi+LD_BPB+BPB_TotalSec16]
00007B8C 668B4619            <1>       mov    ax, [esi+LD_BPB+BPB_TotalSec16]
00007B90 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
00007B93 7503                <1>       jnz    short loc_set_hd_FAT_cluster_count
                             <1> loc_set_32bit_FAT_total_sectors:
00007B95 8B4626              <1>       mov    eax, [esi+LD_BPB+BPB_TotalSec32]
                             <1>       ;mov   [esi+LD_TotalSectors], eax
                             <1> loc_set_hd_FAT_cluster_count:
00007B98 894670              <1>       mov    [esi+LD_TotalSectors], eax ; 14/07/2020
```

```
00007B9B 8B5668              <1>       mov    edx, [esi+LD_DATABegin]
00007B9E 2B566C              <1>       sub    edx, [esi+LD_StartSector]
00007BA1 29D0                <1>       sub    eax, edx
00007BA3 31D2                <1>       xor    edx, edx ; 0
00007BA5 0FB64E13            <1>         movzx   ecx, byte [esi+LD_BPB+BPB_SecPerClust]
00007BA9 F7F1                <1>         div ecx
00007BAB 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
00007BAE E855010000          <1>       call   get_free_FAT_sectors
00007BB3 720D                <1>       jc     short loc_validate_hd_FAT_partition_retn
00007BB5 894674              <1>       mov    [esi+LD_FreeSectors], eax
00007BB8 C6467E06            <1>       mov    byte [esi+LD_MediaChanged], 6  ; Volume Name Reset
                             <1>
                             <1>       ; 15/07/2020
00007BBC FE05[22380100]      <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:
00007BC2 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
00007BC3 8A6604              <1>       mov    ah, [esi+ptFileSystemID]
00007BC6 80FCA1              <1>       cmp    ah, 0A1h ; SINGLIX FS1 (trfs1) partition
00007BC9 7549                <1>       jne    short loc_validate_hd_fs_partition_stc_retn
                             <1> loc_set_valid_hd_fs_partition_params:
00007BCB FE05[22380100]      <1>       inc    byte [Last_DOS_DiskNo] ; > 1
00007BD1 30C0                <1>       xor    al, al ; mov al, 0
                             <1>       ;mov   [drv], dl
00007BD3 29DB                <1>       sub    ebx, ebx ; 0
00007BD5 8A3D[22380100]      <1>       mov    bh, [Last_DOS_DiskNo]
00007BDB 81C300010900        <1>       add    ebx, Logical_DOSDisks
00007BE1 C6430102            <1>       mov    byte [ebx+LD_DiskType], 2
00007BE5 885302              <1>       mov    [ebx+LD_PhyDrvNo], dl
                             <1>       ;mov   [ebx+LD_FATType], al ; 0
                             <1>       ;mov   [ebx+LD_FSType], ah
00007BE8 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:
00007BEC 89DD                <1>       mov    ebp, ebx ; 10/01/2016
                             <1>       ;mov   dl, [ebx+LD_PhyDrvNo]
00007BEE A0[5A870100]        <1>       mov    al, [HD_LBA_yes] ; 10/01/2016
00007BF3 884305              <1>       mov    [ebx+LD_LBAYes], al
00007BF6 89DE                <1>       mov    esi, ebx
00007BF8 BB[66870100]        <1>       mov    ebx, DOSBootSectorBuff ; buffer address
00007BFD 08C0                <1>       or     al, al
00007BFF 7515                <1>       jnz    short loc_hd_fs_drv_init_load_bs_lba
                             <1> loc_hd_fs_drv_init_load_bs_chs:
00007C01 8A7601              <1>       mov    dh, [esi+ptBeginHead]
00007C04 668B4E02            <1>       mov    cx, [esi+ptBeginSector]
00007C08 66B80102            <1>       mov    ax, 0201h ; Read 1 sector
                             <1>       ;mov   ebx, DOSBootSectorBuff
00007C0C E80ED6FFFF          <1>       call   int13h
00007C11 7311                <1>       jnc    short loc_hd_drv_fs_boot_validation
                             <1> loc_validate_hd_fs_partition_err_retn:
00007C13 C3                  <1>       retn
                             <1> loc_validate_hd_fs_partition_stc_retn:
00007C14 F9                  <1>       stc
00007C15 C3                  <1>       retn
                             <1> loc_hd_fs_drv_init_load_bs_lba:
                             <1>       ; DL = Physical drive number
                             <1>       ;mov   esi, ebx
00007C16 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)
00007C19 B41B                <1>       mov    ah, 1Bh ; LBA read
00007C1B B001                <1>       mov    al, 1 ; sector count
00007C1D E8FDD5FFFF          <1>       call   int13h
00007C22 72EF                <1>       jc     short loc_validate_hd_fs_partition_err_retn
                             <1> loc_hd_drv_fs_boot_validation:
                             <1>       ;mov   esi, DOSBootSectorBuff
00007C24 89DE                <1>       mov    esi, ebx ; Boot sector buffer
00007C26 6681BEFE01000055AA  <1>       cmp    word [esi+BS_Validation], 0AA55h
00007C2F 75E3                <1>       jne    short loc_validate_hd_fs_partition_stc_retn
                             <1>         ;
                             <1>       ;Singlix FS Extensions to TR-DOS (7/6/2009)
00007C31 66817E034653        <1>       cmp    word [esi+bs_FS_Identifier], 'FS' ; 03/02/2018
00007C37 75DB                <1>       jne    short loc_validate_hd_fs_partition_stc_retn
```

```
878                                     <1>         ;'A1h' check is not necessary
879                                     <1>       ;  if 'FS' check is passed as OK/Yes.
880 00007C39 807E09A1                   <1>       cmp   byte [esi+bs_FS_PartitionID], 0A1h
881 00007C3D 75D5                       <1>       jne   short loc_validate_hd_fs_partition_stc_retn
882                                     <1>       ;
883 00007C3F 89EF                       <1>       mov   edi, ebp ; 10/01/2016
884                                     <1>       ;
885 00007C41 8A462D                     <1>       mov   al, byte [esi+bs_FS_LBA_Ready]
886 00007C44 884705                     <1>       mov   [edi+LD_FS_LBAYes], al
887                                     <1>       ;
888                                     <1>       ; 03/01/2010 CHS -> DOS FAT/BPB compatibility fix
889 00007C47 8A4608                     <1>       mov   al, [esi+bs_FS_MediaAttrib]
890 00007C4A 884706                     <1>       mov   byte [edi+LD_FS_MediaAttrib], al
891                                     <1>       ;
892 00007C4D 8A460A                     <1>       mov   al, [esi+bs_FS_VersionMaj]
893 00007C50 884707                     <1>       mov   [edi+LD_FS_VersionMajor], al
894                                     <1>       ;
895 00007C53 668B4606                   <1>       mov   ax, [esi+bs_FS_BytesPerSec]
896 00007C57 66894711                   <1>       mov   [edi+LD_FS_BytesPerSec], ax
897 00007C5B 8A462E                     <1>       mov   al, [esi+bs_FS_SecPerTrack]
898 00007C5E 30E4                       <1>       xor   ah, ah ; 09/12/2017
899 00007C60 6689471E                   <1>       mov   [edi+LD_FS_SecPerTrack], ax
900 00007C64 8A462F                     <1>       mov   al, [esi+bs_FS_Heads]
901 00007C67 66894720                   <1>       mov   [edi+LD_FS_NumHeads], ax
902                                     <1>       ;
903 00007C6B 8B4628                     <1>       mov   eax, [esi+bs_FS_UnDelDirD]
904 00007C6E 894722                     <1>       mov   [edi+LD_FS_UnDelDirD], eax
905 00007C71 8B5618                     <1>       mov   edx, [esi+bs_FS_MATLocation]
906 00007C74 89570C                     <1>       mov   [edi+LD_FS_MATLocation], edx
907 00007C77 8B461C                     <1>       mov   eax, [esi+bs_FS_RootDirD]
908 00007C7A 894708                     <1>       mov   [edi+LD_FS_RootDirD], eax
909 00007C7D 8B460C                     <1>       mov   eax, [esi+bs_FS_BeginSector]
910 00007C80 89476C                     <1>       mov   [edi+LD_FS_BeginSector], eax
911 00007C83 8B4710                     <1>       mov   eax, [edi+bs_FS_VolumeSize]
912 00007C86 894770                     <1>       mov   [edi+LD_FS_VolumeSize], eax
913                                     <1>       ;
914 00007C89 89D0                       <1>       mov   eax, edx ; [edi+LD_FS_MATLocation]
915 00007C8B 03476C                     <1>       add   eax, [edi+LD_FS_BeginSector]
916 00007C8E 89FE                       <1>       mov   esi, edi
917                                     <1> mread_hd_fs_MAT_sector:
918                                     <1>        ;mov  ebx, DOSBootSectorBuff
919 00007C90 B901000000                 <1>       mov   ecx, 1
920 00007C95 E883A60000                 <1>       call  disk_read
921 00007C9A 7248                       <1>       jc    short loc_validate_hd_fs_partition_retn
922                                     <1>       ; EDI will not be changed
923 00007C9C 89DE                       <1>       mov   esi, ebx
924                                     <1> use_hdfs_mat_sector_params:
925 00007C9E 8B460C                     <1>       mov   eax, [esi+FS_MAT_DATLocation]
926 00007CA1 894714                     <1>       mov   [edi+LD_FS_DATLocation], eax
927 00007CA4 8B4610                     <1>       mov   eax, [esi+FS_MAT_DATScount]
928 00007CA7 894718                     <1>       mov   [edi+LD_FS_DATSectors], eax
929 00007CAA 8B4614                     <1>       mov   eax, [esi+FS_MAT_FreeSectors]
930 00007CAD 894774                     <1>         mov     [edi+LD_FS_FreeSectors], eax
931 00007CB0 8B4618                     <1>       mov   eax, [esi+FS_MAT_FirstFreeSector]
932 00007CB3 894778                     <1>       mov   [edi+LD_FS_FirstFreeSector], eax
933 00007CB6 8B4708                     <1>       mov   eax, [edi+LD_FS_RootDirD]
934 00007CB9 03476C                     <1>       add   eax, [edi+LD_FS_BeginSector]
935 00007CBC 89FE                       <1>       mov   esi, edi
936                                     <1> read_hd_fs_RDT_sector:
937 00007CBE BB[66870100]               <1>       mov   ebx, DOSBootSectorBuff
938                                     <1>       ;mov  ecx, 1
939 00007CC3 B101                       <1>       mov   cl, 1
940 00007CC5 E853A60000                 <1>       call  disk_read
941 00007CCA 7218                       <1>       jc    short loc_validate_hd_fs_partition_retn
942                                     <1>       ; EDI will not be changed
943 00007CCC 89DE                       <1>       mov   esi, ebx
944                                     <1> use_hdfs_RDT_sector_params:
945 00007CCE 8B461C                     <1>       mov   eax, [esi+FS_RDT_VolumeSerialNo]
946 00007CD1 894728                     <1>       mov   [edi+LD_FS_VolumeSerial], eax
947 00007CD4 57                         <1>       push  edi
948                                     <1>       ;mov  ecx, 16
949 00007CD5 B110                       <1>       mov   cl, 16
950 00007CD7 83C640                     <1>       add   esi, FS_RDT_VolumeName
951 00007CDA 83C72C                     <1>       add   edi, LD_FS_VolumeName
952 00007CDD F3A5                       <1>       rep   movsd ; 64 bytes
953 00007CDF 5E                         <1>       pop   esi
954                                     <1>             ; Volume Name Reset
955 00007CE0 C6467E06                   <1>         mov     byte [esi+LD_FS_MediaChanged], 6
956                                     <1>       ;
957                                     <1>         ;mov cl, [Last_DOS_DiskNo]
958                                     <1>       ;add  cl, 'A'
959                                     <1>       ;mov  [esi+LD_FS_Name], cl
960                                     <1>
961                                     <1> loc_validate_hd_fs_partition_retn:
962 00007CE4 C3                         <1>       retn
963                                     <1>
964                                     <1> load_masterboot:
965                                     <1>       ; 14/07/2020 (Reset function has been removed)
966                                     <1>       ;
967                                     <1>       ; 10/01/2016 (TRDOS 386 = TRDOS v2.0)
968                                     <1>       ; 2005 - 2011
969                                     <1>       ; input -> DL = drive number
970                                     <1> ;     mov   ah, 0Dh ; Alternate disk reset
971                                     <1> ;     call  int13h
972                                     <1> ;     jnc   short pass_reset_error
973                                     <1> ;harddisk_error:
974                                     <1> ;     retn
975                                     <1> ;pass_reset_error:
976 00007CE5 BB[56840100]               <1>       mov   ebx, MasterBootBuff
977 00007CEA 66B80102                   <1>       mov   ax, 0201h
978 00007CEE 66B90100                   <1>       mov   cx, 1
979 00007CF2 30F6                       <1>       xor   dh, dh
980 00007CF4 E826D5FFFF                 <1>       call  int13h
981 00007CF9 720C                       <1>       jc    short harddisk_error
982                                     <1>       ;
```

```
 983 00007CFB 66813D[54860100]55- <1>       cmp    word [MBIDCode], 0AA55h
 983 00007D03 AA                  <1>
 984 00007D04 7401                <1>       je     short load_masterboot_ok
 985 00007D06 F9                  <1>       stc
 986                              <1> harddisk_error:
 987                              <1> load_masterboot_ok:
 988 00007D07 C3                  <1>       retn
 989                              <1>
 990                              <1> get_free_FAT_sectors:
 991                              <1>       ; 21/12/2017
 992                              <1>       ; 29/02/2016
 993                              <1>       ; 13/02/2016
 994                              <1>       ; 04/02/2016
 995                              <1>       ; 07/01/2016 (TRDOS 386 = TRDOS v2.0)
 996                              <1>       ; 11/07/2010
 997                              <1>       ; 21/06/2009
 998                              <1>       ; INPUT: ESI = Logical DOS Drive Description Table address
 999                              <1>       ; OUTPUT: STC => Error
1000                              <1>         ;    cf = 0 and EAX = Free FAT sectors
1001                              <1>       ; Also, related parameters and FAT buffer will be reset and updated
1002                              <1>
1003 00007D08 31C0                <1>       xor    eax, eax
1004                              <1>       ;mov   [esi+LD_FreeSectors], eax ; Reset
1005                              <1>
1006 00007D0A 807E0302            <1>         cmp     byte [esi+LD_FATType], 2
1007 00007D0E 7654                <1>       jna    short loc_gfc_get_fat_free_clusters
1008                              <1>
1009                              <1>       ; 29/02/2016
1010 00007D10 48                  <1>       dec    eax ; 0FFFFFFFFh
1011 00007D11 89463A              <1>       mov    [esi+LD_BPB+BPB_Reserved], eax ; Free cluster count (reset)
1012 00007D14 89463E              <1>       mov    [esi+LD_BPB+BPB_Reserved+4], eax ; First Free Cluster (reset)
1013 00007D17 40                  <1>       inc    eax ; 0
1014                              <1>       ;
1015 00007D18 668B4636            <1>       mov    ax, [esi+LD_BPB+BPB_FSInfo]
1016 00007D1C 03466C              <1>       add    eax, [esi+LD_StartSector]
1017                              <1>
1018 00007D1F BB[66870100]        <1>       mov    ebx, DOSBootSectorBuff
1019 00007D24 B901000000          <1>       mov    ecx, 1
1020 00007D29 E8EFA50000          <1>       call   disk_read
1021 00007D2E 7301                <1>       jnc    short loc_gfc_check_fsinfo_signs
1022                              <1> retn_gfc_get_fsinfo_sec:
1023 00007D30 C3                  <1>       retn
1024                              <1>
1025                              <1> loc_gfc_check_fsinfo_signs:
1026 00007D31 BB[66870100]        <1>       mov    ebx, DOSBootSectorBuff ; 13/02/2016
1027 00007D36 813B52526141        <1>         cmp     dword [ebx], 41615252h
1028 00007D3C 7524                <1>       jne    short retn_gfc_get_fsinfo_stc
1029                              <1>       ;add   ebx, 484
1030                              <1>       ;cmp   dword [ebx], 61417272h
1031 00007D3E 81BBE4010000727241- <1>       cmp    dword [ebx+484], 61417272h
1031 00007D47 61                  <1>
1032 00007D48 7518                <1>       jne    short retn_gfc_get_fsinfo_stc
1033                              <1>       ;add   ebx, 4
1034                              <1>       ;mov   eax, [ebx]
1035 00007D4A 8B83E8010000        <1>       mov    eax, [ebx+488]
1036                              <1>       ; 29/02/2016
1037 00007D50 89463A              <1>       mov    [esi+LD_BPB+BPB_Reserved], eax ; Free cluster count
1038 00007D53 8B93EC010000        <1>       mov    edx,  [ebx+492]
1039 00007D59 89463E              <1>       mov    [esi+LD_BPB+BPB_Reserved+4], eax ; First Free Cluster
1040                              <1>       ; 21/12/2017
1041 00007D5C 89C3                <1>       mov    ebx, eax ; (initial value = 0FFFFFFFFh)
1042 00007D5E 43                  <1>       inc    ebx ; 0FFFFFFFFh -> 0
1043 00007D5F 7513                <1>       jnz    short short retn_from_get_free_fat32_clusters
1044 00007D61 C3                  <1>       retn
1045                              <1>
1046                              <1> retn_gfc_get_fsinfo_stc:
1047 00007D62 F9                  <1>       stc
1048 00007D63 C3                  <1>       retn
1049                              <1>
1050                              <1> loc_gfc_get_fat_free_clusters:
1051                              <1>       ;mov   eax, 2
1052 00007D64 B002                <1>       mov    al, 2
1053                              <1>       ;mov   [FAT_CurrentCluster], eax
1054                              <1> loc_gfc_loop_get_next_cluster:
1055 00007D66 E8EB4F0000          <1>       call   get_next_cluster
1056 00007D6B 730E                <1>       jnc    short loc_gfc_free_fat_clusters_cont
1057 00007D6D 21C0                <1>       and    eax, eax
1058 00007D6F 7411                <1>       jz     short loc_gfc_pass_inc_free_cluster_count
1059                              <1>
1060                              <1> retn_from_get_free_fat_clusters:
1061 00007D71 8B4674              <1>       mov    eax, [esi+LD_FreeSectors] ; Free clusters !
1062                              <1> retn_from_get_free_fat32_clusters:
1063 00007D74 0FB65E13            <1>         movzx      ebx, byte [esi+LD_BPB+BPB_SecPerClust]
1064 00007D78 F7E3                <1>             mul    ebx
1065                              <1>       ;mov   [esi+LD_FreeSectors], eax ; Free sectors
1066                              <1> retn_get_free_sectors_calc:
1067 00007D7A C3                  <1>       retn
1068                              <1>
1069                              <1> loc_gfc_free_fat_clusters_cont:
1070 00007D7B 09C0                <1>       or     eax, eax
1071 00007D7D 7503                <1>       jnz    short loc_gfc_pass_inc_free_cluster_count
1072 00007D7F FF4674              <1>       inc    dword [esi+LD_FreeSectors] ; Free clusters !
1073                              <1>
1074                              <1> loc_gfc_pass_inc_free_cluster_count:
1075                              <1>       ;mov   eax, [FAT_CurrentCluster]
1076 00007D82 89C8                <1>       mov    eax, ecx ; [FAT_CurrentCluster]
1077 00007D84 3B4678              <1>       cmp    eax, [esi+LD_Clusters]
1078 00007D87 77E8                <1>       ja     short retn_from_get_free_fat_clusters
1079 00007D89 40                  <1>       inc    eax
1080                              <1>       ;mov   [FAT_CurrentCluster], eax
1081 00007D8A EBDA                <1>       jmp    short loc_gfc_loop_get_next_cluster
1082                              <1>
1083                              <1> floppy_drv_init:
1084                              <1>       ; 09/12/2017
1085                              <1>       ; 06/07/2016
```

```
1086                            <1>       ; 10/01/2016 (TRDOS 386 = TRDOS v2.0)
1087                            <1>       ; 24/07/2011
1088                            <1>       ; 04/07/2009
1089                            <1>       ; INPUT ->
1090                            <1>       ;      DL = Drive number (0,1)
1091                            <1>       ; OUTPUT ->
1092                            <1>       ;      BL = drive name
1093                            <1>       ;      BH = drive number
1094                            <1>       ;      ESI = Logical DOS drv description table
1095                            <1>       ;      EAX = Volume serial number
1096                            <1>
1097 00007D8C BE[EE6D0000]      <1>       mov    esi, fd0_type ; 10/01/2016
1098 00007D91 BF00010900        <1>       mov    edi, Logical_DOSDisks
1099 00007D96 08D2              <1>       or     dl, dl
1100 00007D98 7407              <1>       jz     short loc_drv_init_fd0_fd1
1101 00007D9A 81C700010000      <1>       add    edi, 100h
1102 00007DA0 46                <1>       inc    esi ; fd1_type ; 10/01/2016
1103                            <1> loc_drv_init_fd0_fd1:
1104 00007DA1 C6477E00          <1>       mov    byte [edi+LD_MediaChanged], 0
1105 00007DA5 803E01            <1>       cmp    byte [esi], 1 ; type (>0 if it is existing)
1106                            <1>              ; 4 = 1.44 MB, 80 track, 3 1/2"
1107 00007DA8 7221              <1>       jb     short read_fd_boot_sector_retn
1108 00007DAA 885702            <1>       mov    [edi+LD_PhyDrvNo], dl
1109                            <1> read_fd_boot_sector:
1110 00007DAD 30F6              <1>       xor    dh, dh
1111 00007DAF B904000000        <1>       mov    ecx, 4 ; Retry Count
1112                            <1> read_fd_boot_sector_again:
1113 00007DB4 51                <1>       push   ecx
1114                            <1>       ;mov   cx, 1
1115 00007DB5 B101              <1>       mov    cl, 1
1116 00007DB7 66B80102          <1>       mov    ax, 0201h ; Read 1 sector
1117 00007DBB BB[66870100]      <1>       mov    ebx, DOSBootSectorBuff
1118 00007DC0 E85AD4FFFF        <1>       call   int13h
1119 00007DC5 59                <1>       pop    ecx
1120 00007DC6 7304              <1>       jnc    short use_fd_boot_sector_params
1121 00007DC8 E2EA              <1>       loop   read_fd_boot_sector_again
1122                            <1>
1123                            <1> read_fd_boot_sector_stc_retn:
1124 00007DCA F9                <1>       stc
1125                            <1> read_fd_boot_sector_retn:
1126 00007DCB C3                <1>       retn
1127                            <1>
1128                            <1> use_fd_boot_sector_params:
1129                            <1>       ;mov   esi, DOSBootSectorBuff
1130 00007DCC 89DE              <1>       mov    esi, ebx
1131 00007DCE 6681BEFE01000055AA <1>      cmp    word [esi+BS_Validation], 0AA55h
1132 00007DD7 75F1              <1>       jne    short read_fd_boot_sector_stc_retn
1133 00007DD9 66817E035346      <1>         cmp     word [esi+bs_FS_Identifier], 'SF'
1134 00007DDF 0F85A2000000      <1>         jne     use_fd_fatfs_boot_sector_params
1135                            <1>       ;
1136 00007DE5 8A462D            <1>       mov    al, [esi+bs_FS_LBA_Ready]
1137 00007DE8 884705            <1>       mov    [edi+LD_FS_LBAYes], al
1138                            <1>       ;
1139                            <1>       ; 03/01/2010 CHS -> DOS FAT/BPB compatibility fix
1140 00007DEB 8A4608            <1>       mov    al, [esi+bs_FS_MediaAttrib]
1141 00007DEE 884706            <1>       mov    [edi+LD_FS_MediaAttrib], al
1142                            <1>       ;
1143 00007DF1 8A460A            <1>         mov  al, [esi+bs_FS_VersionMaj]
1144 00007DF4 884707            <1>       mov    byte [edi+LD_FS_VersionMajor], al
1145 00007DF7 668B4606          <1>       mov    ax, [esi+bs_FS_BytesPerSec]
1146 00007DFB 66894711          <1>       mov    [edi+LD_FS_BytesPerSec], ax
1147 00007DFF 8A462E            <1>       mov    al, [esi+bs_FS_SecPerTrack]
1148 00007E02 28E4              <1>       sub    ah, ah ; 09/12/2017
1149 00007E04 6689471E          <1>       mov    [edi+LD_FS_SecPerTrack], ax
1150 00007E08 8A462F            <1>       mov    al, [esi+bs_FS_Heads]
1151 00007E0B 66894720          <1>       mov    [edi+LD_FS_NumHeads], ax
1152                            <1>       ;
1153 00007E0F 8B4628            <1>       mov    eax, [esi+bs_FS_UnDelDirD]
1154 00007E12 894722            <1>       mov    [edi+LD_FS_UnDelDirD], eax
1155 00007E15 8B4618            <1>       mov    eax, [esi+bs_FS_MATLocation]
1156 00007E18 89470C            <1>       mov    [edi+LD_FS_MATLocation], eax
1157 00007E1B 8B461C            <1>       mov    eax, [esi+bs_FS_RootDirD]
1158 00007E1E 894708            <1>       mov    [edi+LD_FS_RootDirD], eax
1159 00007E21 8B460C            <1>       mov    eax, [esi+bs_FS_BeginSector]
1160 00007E24 89476C            <1>       mov    [edi+LD_FS_BeginSector], eax
1161 00007E27 8B4610            <1>       mov    eax, [esi+bs_FS_VolumeSize]
1162 00007E2A 894770            <1>       mov    [edi+LD_FS_VolumeSize], eax
1163                            <1>       ;
1164 00007E2D 89FE              <1>       mov    esi, edi
1165 00007E2F 8B460C            <1>       mov    eax, [esi+LD_FS_MATLocation]
1166                            <1>       ;add   eax, [edi+LD_FS_BeginSector]
1167                            <1> read_fd_MAT_sector_again:
1168                            <1>       ;mov   ebx, DOSBootSectorBuff
1169                            <1>       ;mov   ecx, 1
1170 00007E32 B101              <1>       mov    cl, 1
1171 00007E34 E8EAA40000        <1>       call   chs_read
1172 00007E39 89DE              <1>       mov    esi, ebx
1173 00007E3B 7301              <1>       jnc    short use_fdfs_mat_sector_params
1174                            <1>       ;jmp   short read_fd_boot_sector_retn
1175 00007E3D C3                <1>       retn
1176                            <1> use_fdfs_mat_sector_params:
1177 00007E3E 8B460C            <1>       mov    eax, [esi+FS_MAT_DATLocation]
1178 00007E41 894714            <1>       mov    [edi+LD_FS_DATLocation], eax
1179 00007E44 8B4610            <1>       mov    eax, [esi+FS_MAT_DATScount]
1180 00007E47 894718            <1>       mov    [edi+LD_FS_DATSectors], eax
1181 00007E4A 8B4714            <1>       mov    eax, [edi+FS_MAT_FreeSectors]
1182 00007E4D 894774            <1>       mov    [edi+LD_FS_FreeSectors], eax
1183 00007E50 8B4618            <1>       mov    eax, [esi+FS_MAT_FirstFreeSector]
1184 00007E53 894778            <1>       mov    [edi+LD_FS_FirstFreeSector], eax
1185                            <1>       ;
1186 00007E56 89FE              <1>       mov    esi, edi
1187 00007E58 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 00007E5B B101              <1>       mov    cl, 1
1192 00007E5D E8C1A40000        <1>       call   chs_read
1193 00007E62 89DE              <1>       mov    esi, ebx
1194 00007E64 7220              <1>       jc     short read_fd_RDT_sector_retn
1195                            <1> use_fdfs_RDT_sector_params:
1196 00007E66 8B461C            <1>       mov    eax, [esi+FS_RDT_VolumeSerialNo]
1197 00007E69 894728            <1>       mov    [edi+LD_FS_VolumeSerial], eax
1198 00007E6C 57                <1>       push   edi
1199                            <1>       ;mov   ecx, 16
1200 00007E6D B110              <1>       mov    cl, 16
1201 00007E6F 83C640            <1>       add    esi, FS_RDT_VolumeName
1202 00007E72 83C72C            <1>       add    edi, LD_FS_VolumeName
1203 00007E75 F3A5              <1>       rep    movsd ; 64 bytes
1204 00007E77 5E                <1>       pop    esi
1205 00007E78 C6460300          <1>       mov    byte [esi+LD_FATType], 0
1206 00007E7C C64604A1          <1>       mov    byte [esi+LD_FSType], 0A1h
1207 00007E80 E9A5000000        <1>         jmp     loc_cont_use_fd_boot_sector_params
1208                            <1>
1209                            <1> read_fd_RDT_sector_stc_retn:
1210 00007E85 F9                <1>       stc
1211                            <1> read_fd_RDT_sector_retn:
1212 00007E86 C3                <1>       retn
1213                            <1>
1214                            <1> use_fd_fatfs_boot_sector_params:
1215 00007E87 807E2629          <1>       cmp    byte [esi+BS_BootSig], 29h
1216 00007E8B 75F8              <1>       jne    short read_fd_RDT_sector_stc_retn
1217 00007E8D 807E15F0          <1>       cmp    byte [esi+BPB_Media], 0F0h
1218 00007E91 72F3              <1>       jb     short read_fd_RDT_sector_retn
1219 00007E93 57                <1>       push   edi
1220 00007E94 83C706            <1>       add    edi, LD_BPB
1221                            <1>       ;mov   ecx, 16
1222 00007E97 B110              <1>       mov    cl, 16
1223 00007E99 F3A5              <1>       rep    movsd ; 64 bytes
1224 00007E9B 5E                <1>       pop    esi
1225 00007E9C 31C0              <1>       xor    eax, eax
1226 00007E9E 89466C            <1>       mov    [esi+LD_StartSector], eax ; 0
1227 00007EA1 668B461C          <1>       mov    ax, [esi+LD_BPB+BPB_FATSz16]
1228 00007EA5 8A4E16            <1>       mov    cl, [esi+LD_BPB+BPB_NumFATs]
1229 00007EA8 F7E1              <1>       mul    ecx
1230                            <1>       ; edx = 0 !
1231 00007EAA 668B5614          <1>       mov    dx, [esi+LD_BPB+BPB_RsvdSecCnt]
1232 00007EAE 66895660          <1>       mov    [esi+LD_FATBegin], dx
1233                            <1>       ;add   eax, edx
1234 00007EB2 6601D0            <1>       add    ax, dx
1235 00007EB5 894664            <1>       mov    [esi+LD_ROOTBegin], eax
1236 00007EB8 894668            <1>       mov    [esi+LD_DATABegin], eax
1237 00007EBB 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 00007EBF 6683C20F          <1>       add    dx, 15 ; 06/07/2016 (+(512/32)-1)
1245 00007EC3 66C1EA04          <1>       shr    dx, 4 ; / 16 (==16 entries per sector)
1246 00007EC7 015668            <1>       add    [esi+LD_DATABegin], edx ; + rd sectors
1247                            <1>       ;movzx eax, word [esi+LD_BPB+BPB_TotalSec16]
1248 00007ECA 668B4619          <1>       mov    ax, [esi+LD_BPB+BPB_TotalSec16]
1249 00007ECE 894670            <1>       mov    [esi+LD_TotalSectors], eax
1250 00007ED1 2B4668            <1>       sub    eax, [esi+LD_DATABegin]
1251                            <1>       ;movzx ecx, byte [esi+LD_BPB+BPB_SecPerClust]
1252 00007ED4 8A4E13            <1>       mov    cl, [esi+LD_BPB+BPB_SecPerClust]
1253 00007ED7 80F901            <1>       cmp    cl, 1
1254 00007EDA 7605              <1>       jna    short save_fd_fatfs_cluster_count
1255                            <1>       ;sub   edx, edx
1256 00007EDC 6629D2            <1>       sub    dx, dx ; 0
1257                            <1>       ;sub   dl, dl ; 06/07/2016
1258 00007EDF F7F1              <1>       div    ecx
1259                            <1> save_fd_fatfs_cluster_count:
1260 00007EE1 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 00007EE4 29C0              <1>       sub    eax, eax ; 0
1267 00007EE6 A2[6A890100]      <1>       mov    [FAT_BuffValidData], al ; 0
1268 00007EEB A2[6B890100]      <1>       mov    [FAT_BuffDrvName], al ; 0
1269 00007EF0 A3[6E890100]      <1>       mov    [FAT_BuffSector], eax ; 0
1270                            <1>
1271                            <1> read_fd_FAT_sectors:
1272 00007EF5 BB001C0900        <1>       mov    ebx, FAT_Buffer
1273 00007EFA 668B4614          <1>       mov    ax, [esi+LD_BPB+BPB_RsvdSecCnt]
1274                            <1>       ;mov   ecx, 3
1275 00007EFE B103              <1>       mov    cl, 3 ; 3 sectors
1276 00007F00 E81EA40000        <1>       call   chs_read
1277 00007F05 7240              <1>       jc     short read_fd_FAT_sectors_retn
1278                            <1> use_fd_FAT_sectors:
1279 00007F07 8A4602            <1>       mov    al, [esi+LD_PhyDrvNo]
1280 00007F0A 0441              <1>       add    al, 'A'
1281 00007F0C A2[6B890100]      <1>       mov    [FAT_BuffDrvName], al
1282 00007F11 C605[6A890100]01  <1>       mov    byte [FAT_BuffValidData], 1
1283 00007F18 E82B000000        <1>       call   fd_init_calculate_free_clusters
1284 00007F1D 7228              <1>       jc     short read_fd_FAT_sectors_retn
1285                            <1>
1286                            <1> loc_use_fd_boot_sector_params_FAT:
1287 00007F1F C6460301          <1>       mov    byte [esi+LD_FATType], 1 ; FAT 12
1288 00007F23 C6460401          <1>       mov    byte [esi+LD_FSType], 1
1289 00007F27 8B462D            <1>         mov     eax, [esi+LD_BPB+VolumeID]
1290                            <1> loc_cont_use_fd_boot_sector_params:
1291 00007F2A 8A7E02            <1>       mov    bh, [esi+LD_PhyDrvNo]
1292 00007F2D 887E7D            <1>       mov    [esi+LD_DParamEntry], bh
1293 00007F30 88FB              <1>       mov    bl, bh
1294 00007F32 80C341            <1>       add    bl, 'A'
1295 00007F35 881E              <1>       mov    byte [esi+LD_Name], bl
```

```
1296 00007F37 C6460101            <1>       mov   byte [esi+LD_DiskType], 1
1297 00007F3B C6460500            <1>       mov   byte [esi+LD_LBAYes], 0
1298 00007F3F C6467C00            <1>       mov   byte [esi+LD_PartitionEntry], 0
1299 00007F43 C6467E06            <1>       mov   byte [esi+LD_MediaChanged], 6 ; Volume Name Reset
1300                              <1>
1301                              <1> read_fd_FAT_sectors_retn:
1302 00007F47 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 00007F48 29C0                <1>       sub   eax, eax
1314 00007F4A 894674              <1>       mov   [esi+LD_FreeSectors], eax ; 0
1315 00007F4D B002                <1>       mov   al, 2 ; eax = 2
1316                              <1>
1317                              <1> fd_init_loop_get_next_cluster:
1318 00007F4F E830000000          <1>       call  fd_init_get_next_cluster
1319 00007F54 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 00007F56 6621C0              <1>       and   ax, ax
1327 00007F59 7504                <1>       jnz   short fd_init_pass_inc_free_cluster_count
1328                              <1>       ;inc  dword [esi+LD_FreeSectors]
1329 00007F5B 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 00007F5F 66A1[66890100]      <1>       mov   ax, [FAT_CurrentCluster]
1334                              <1>       ;cmp  eax, [esi+LD_Clusters]
1335 00007F65 663B4678            <1>       cmp   ax, [esi+LD_Clusters]
1336 00007F69 7704                <1>       ja    short short retn_from_fd_init_calculate_free_clusters
1337                              <1>       ;inc  eax
1338 00007F6B 6640                <1>       inc   ax
1339 00007F6D EBE0                <1>       jmp   short fd_init_loop_get_next_cluster
1340                              <1>
1341                              <1> retn_from_fd_init_calculate_free_clusters:
1342 00007F6F 8A4613              <1>       mov   al, [esi+LD_BPB+BPB_SecPerClust]
1343 00007F72 3C01                <1>       cmp   al, 1
1344 00007F74 760D                <1>       jna   short fd_init_calculate_free_clusters_retn
1345                              <1>       ;movzx eax, al
1346 00007F76 30E4                <1>       xor   ah, ah ; 09/12/2017
1347                              <1>       ;mov  ecx, [esi+LD_FreeSectors]
1348 00007F78 668B4E74            <1>       mov   cx, [esi+LD_FreeSectors] ; Count of free clusters
1349                              <1>       ;mul  ecx
1350 00007F7C 66F7E1              <1>       mul   cx
1351                              <1>       ;mov  [esi+LD_FreeSectors], eax
1352 00007F7F 66894674            <1>       mov   [esi+LD_FreeSectors], ax
1353                              <1> fd_init_calculate_free_clusters_retn:
1354 00007F83 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 00007F84 A3[66890100]        <1>       mov   [FAT_CurrentCluster], eax
1369                              <1> fd_init_get_next_cluster_readnext:
1370 00007F89 29D2                <1>       sub   edx, edx ; 0
1371 00007F8B BB00040000          <1>       mov   ebx, 1024 ; 400h
1372 00007F90 F7F3                <1>       div   ebx
1373                              <1>       ; EAX = Count of 3 FAT sectors
1374                              <1>       ; EDX = Buffer entry index
1375 00007F92 89C1                <1>       mov   ecx, eax
1376                              <1>       ;mov  eax, 3
1377 00007F94 B003                <1>       mov   al, 3
1378 00007F96 F7E2                <1>       mul   edx ; Multiply by 3
1379 00007F98 66D1E8              <1>       shr   ax, 1 ; Divide by 2
1380 00007F9B 89C3                <1>       mov   ebx, eax ; Buffer byte offset
1381 00007F9D 81C3001C0900        <1>       add   ebx, FAT_Buffer
1382 00007FA3 89C8                <1>       mov   eax, ecx
1383                              <1>       ;mov  edx, 3
1384 00007FA5 66BA0300            <1>       mov   dx, 3
1385 00007FA9 F7E2                <1>       mul   edx
1386                              <1>       ; EAX = FAT Beginning Sector
1387                              <1>       ; EDX = 0
1388 00007FAB 8A0E                <1>       mov   cl, [esi+LD_Name]
1389                              <1>       ;cmp  byte [FAT_BuffValidData], 0
1390                              <1>       ;jna  short fd_init_load_FAT_sectors0
1391 00007FAD 3A0D[6B890100]      <1>       cmp   cl, [FAT_BuffDrvName]
1392 00007FB3 751E                <1>       jne   short fd_init_load_FAT_sectors0
1393 00007FB5 3B05[6E890100]      <1>       cmp   eax, [FAT_BuffSector]
1394 00007FBB 751C                <1>       jne   short fd_init_load_FAT_sectors1
1395                              <1>       ;mov  eax, [FAT_CurrentCluster]
1396 00007FBD A0[66890100]        <1>       mov   al, [FAT_CurrentCluster]
1397                              <1>       ;shr  eax, 1
1398 00007FC2 D0E8                <1>       shr   al, 1
1399 00007FC4 668B03              <1>       mov   ax, [ebx]
1400 00007FC7 7306                <1>       jnc   short fd_init_gnc_even
```

```
1401 00007FC9 66C1E804            <1>       shr    ax, 4
1402                              <1> fd_init_gnc_clc_retn:
1403 00007FCD F8                  <1>       clc
1404 00007FCE C3                  <1>       retn
1405                              <1>
1406                              <1> fd_init_gnc_even:
1407 00007FCF 80E40F              <1>       and    ah, 0Fh
1408 00007FD2 C3                  <1>       retn
1409                              <1>
1410                              <1> fd_init_load_FAT_sectors0:
1411 00007FD3 880D[6B890100]      <1>       mov    [FAT_BuffDrvName], cl
1412                              <1> fd_init_load_FAT_sectors1:
1413 00007FD9 C605[6A890100]00    <1>       mov    byte [FAT_BuffValidData], 0
1414 00007FE0 A3[6E890100]        <1>       mov    [FAT_BuffSector], eax
1415 00007FE5 034660              <1>       add    eax, [esi+LD_FATBegin]
1416 00007FE8 BB001C0900          <1>       mov    ebx, FAT_Buffer
1417                              <1>       ;movzx ecx, word [esi+LD_BPB+BPB_FATSz16]
1418 00007FED 668B4E1C            <1>       mov    cx, [esi+LD_BPB+BPB_FATSz16]
1419 00007FF1 662B0D[6E890100]    <1>       sub    cx, [FAT_BuffSector]
1420                              <1>         ;cmp ecx, 3
1421 00007FF8 6683F903            <1>       cmp    cx, 3
1422 00007FFC 7605                <1>       jna    short fdinit_pass_fix_sector_count_3
1423                              <1>       ;mov   ecx, 3
1424 00007FFE B903000000          <1>       mov    ecx, 3
1425                              <1> fdinit_pass_fix_sector_count_3:
1426 00008003 E81BA30000          <1>       call   chs_read
1427 00008008 730D                <1>       jnc    short fd_init_FAT_sectors_no_load_error
1428 0000800A C605[6A890100]00    <1>       mov    byte [FAT_BuffValidData], 0
1429                              <1>              ; Drv not ready or read Error !
1430 00008011 B80F000000          <1>       mov    eax, ERR_DRV_NOT_RDY ; 15
1431                              <1>       ;xor   edx, edx
1432 00008016 C3                  <1>       retn
1433                              <1>
1434                              <1> fd_init_FAT_sectors_no_load_error:
1435 00008017 C605[6A890100]01    <1>       mov    byte [FAT_BuffValidData], 1
1436 0000801E A1[66890100]        <1>       mov    eax, [FAT_CurrentCluster]
1437 00008023 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 00008028 89DE                <1>       mov    esi, ebx
1455 0000802A 81E600FF0000        <1>       and    esi, 0FF00h ; esi = bh
1456 00008030 81C600010900        <1>       add    esi, Logical_DOSDisks
1457 00008036 8A06                <1>       mov     al, [esi+LD_Name]
1458 00008038 8A6603              <1>       mov     ah, [esi+LD_FATType]
1459 0000803B 80FC01              <1>       cmp     ah, 1
1460 0000803E 7210                <1>       jb     short loc_gfvn_dir_load_err
1461 00008040 3C41                <1>       cmp    al, 'A'
1462 00008042 720C                <1>       jb      short loc_gfvn_dir_load_err
1463 00008044 80FC02              <1>       cmp    ah, 2
1464 00008047 7708                <1>       ja      short get_FAT32_root_cluster
1465                              <1>
1466 00008049 E8634E0000          <1>       call    load_FAT_root_directory
1467 0000804E 730B                <1>       jnc     short loc_get_volume_name
1468                              <1>
1469                              <1> loc_gfvn_dir_load_err:
1470 00008050 C3                  <1>       retn
1471                              <1>
1472                              <1> get_FAT32_root_cluster:
1473 00008051 8B4632              <1>       mov    eax, [esi+LD_BPB+BPB_RootClus]
1474 00008054 E8E34E0000          <1>       call    load_FAT_sub_directory
1475 00008059 7224                <1>       jc     short loc_get_volume_name_retn
1476                              <1>
1477                              <1> loc_get_volume_name:
1478 0000805B BE00000800          <1>         mov     esi, Directory_Buffer
1479 00008060 6631C9              <1>       xor    cx, cx ; 0
1480                              <1> check_root_volume_name:
1481 00008063 8A06                <1>       mov    al, [esi]
1482 00008065 08C0                <1>       or      al, al
1483 00008067 7416                <1>       jz      short loc_get_volume_name_retn
1484 00008069 807E0B08            <1>       cmp     byte [esi+0Bh], 08h
1485 0000806D 7410                <1>       je      short loc_get_volume_name_retn
1486 0000806F 663B0D[7F890100]    <1>       cmp     cx, [DirBuff_LastEntry]
1487 00008076 7308                <1>       jnb     short pass_check_root_volume_name
1488 00008078 6641                <1>       inc     cx
1489 0000807A 83C620              <1>       add     esi, 32
1490 0000807D EBE4                <1>       jmp     short check_root_volume_name
1491                              <1>
1492                              <1> loc_get_volume_name_retn:
1493 0000807F C3                  <1>       retn
1494                              <1>
1495                              <1> pass_check_root_volume_name:
1496 00008080 803D[7B890100]03    <1>       cmp    byte [DirBuff_FATType], 3
1497 00008087 7230                <1>       jb     short loc_get_volume_name_retn_xor
1498                              <1>
1499 00008089 BB001C0900          <1>       mov    ebx, FAT_Buffer
1500 0000808E BE00010900          <1>       mov    esi, Logical_DOSDisks
1501 00008093 31C0                <1>       xor    eax, eax
1502 00008095 8A25[7A890100]      <1>       mov    ah, [DirBuff_DRV]
1503 0000809B 80EC41              <1>       sub    ah, 'A'
1504 0000809E 01C6                <1>       add    esi, eax
1505 000080A0 A1[81890100]        <1>       mov    eax, [DirBuff_Cluster]
```

```
1506 000080A5 E8AC4C0000          <1>       call   get_next_cluster
1507 000080AA 7305                <1>       jnc    short loc_gfvn_load_FAT32_dir_cluster
1508                              <1>
1509 000080AC 83F801              <1>       cmp     eax, 1
1510 000080AF F5                  <1>       cmc
1511 000080B0 C3                  <1>       retn
1512                              <1>
1513                              <1> loc_gfvn_load_FAT32_dir_cluster:
1514 000080B1 E8864E0000          <1>       call   load_FAT_sub_directory
1515 000080B6 73A3                <1>       jnc    short loc_get_volume_name
1516 000080B8 C3                  <1>       retn
1517                              <1>
1518                              <1> loc_get_volume_name_retn_xor:
1519 000080B9 31C0                <1>       xor    eax, eax
1520 000080BB 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 000080BC B416                <1>       mov    ah, 16h
1532 000080BE E85CD1FFFF          <1>       call   int13h
1533 000080C3 80FC06              <1>       cmp    ah, 06h
1534 000080C6 7405                <1>       je     short loc_gmc_status_retn
1535 000080C8 08E4                <1>       or     ah, ah
1536 000080CA 7401                <1>       jz     short loc_gmc_status_retn
1537                              <1> loc_gmc_status_stc_retn:
1538 000080CC F9                  <1>       stc
1539                              <1> loc_gmc_status_retn:
1540 000080CD C3                  <1>       retn
3081                                  %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 000080CE 31DB                <1>       xor    ebx, ebx
  34 000080D0 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 000080D2 BE00010900          <1>       mov    esi, Logical_DOSDisks
  43 000080D7 01DE                <1>       add    esi, ebx
  44                              <1> loc_ccdrv_dos_drive_name_check:
  45 000080D9 80FA02              <1>       cmp    dl, 2
  46 000080DC 720F                <1>       jb     short loc_ccdrv_dos_drive_name_check_ok
  47                              <1>
  48 000080DE 8A06                <1>       mov    al, [esi+LD_Name]
  49 000080E0 2C41                <1>       sub    al, 'A'
  50 000080E2 38D0                <1>       cmp    al, dl
  51 000080E4 7407                <1>       je     short loc_ccdrv_dos_drive_name_check_ok
  52                              <1>
  53                              <1> loc_ccdrv_drive_not_ready_err:
  54                              <1>       ; 16/10/2016 (15h -> 15)
  55 000080E6 B80F000000          <1>       mov    eax, 15 ; Drive not ready
  56                              <1> loc_change_current_drive_stc_retn:
  57 000080EB F9                  <1>       stc
  58 000080EC C3                  <1>       retn
  59                              <1>
  60                              <1> loc_ccdrv_dos_drive_name_check_ok:
  61 000080ED 8A667E              <1>       mov    ah, [esi+LD_MediaChanged]
  62 000080F0 80FC06              <1>       cmp    ah, 6  ; VOLUME NAME CHECK/MOVE SIGN
  63 000080F3 7455                <1>       je     short loc_ccdrv_get_FAT_volume_name_0
  64                              <1>
  65 000080F5 80FA01              <1>       cmp    dl, 1
  66 000080F8 777D                <1>       ja     short loc_gmcs_init_drv_hd
  67                              <1>
  68                              <1> loc_gmcs_init_drv_fd:
  69 000080FA 08E4                <1>       or     ah, ah
```

```
                        <1>       ; AH = 1 is initialization sign (invalid_fd_parameter)
000080FC 7517           <1>       jnz    short loc_ccdrv_call_fd_init
                        <1>
000080FE E8B9FFFFFF     <1>       call   get_media_change_status
00008103 72E1           <1>       jc     short loc_ccdrv_drive_not_ready_err
                        <1>
00008105 20E4           <1>       and    ah, ah
00008107 7476           <1>       jz     short loc_change_current_drv3
                        <1>
00008109 80F406         <1>       xor    ah, 6
0000810C 75D8           <1>       jnz    short loc_ccdrv_drive_not_ready_err
                        <1>
                        <1> loc_ccdrv_call_fd_init_check_vol_id:
0000810E E8440A0000     <1>       call   get_volume_serial_number
00008113 730D           <1>       jnc    short loc_ccdrv_check_vol_serial
                        <1>
                        <1> loc_ccdrv_call_fd_init:
00008115 E872FCFFFF     <1>       call   floppy_drv_init
0000811A 731A           <1>       jnc    short loc_reset_drv_fd_current_dir
                        <1>
                        <1> loc_ccdrv_fdinit_fail_retn:
                        <1>       ; 16/10/2016
0000811C B80F000000     <1>       mov    eax, 15 ; Drive not ready
00008121 C3             <1>       retn
                        <1>
                        <1> loc_ccdrv_check_vol_serial:
00008122 A3[4C820100]   <1>       mov    [Current_VolSerial], eax
                        <1>       ;mov   dl, bh
00008127 E860FCFFFF     <1>       call   floppy_drv_init
0000812C 72EE           <1>       jc     short loc_ccdrv_fdinit_fail_retn
                        <1>
0000812E 3B05[4C820100] <1>       cmp    eax, [Current_VolSerial]
00008134 7445           <1>       je     short loc_change_current_drv2
                        <1>
                        <1> loc_reset_drv_fd_current_dir:
00008136 31C0           <1>       xor    eax, eax
00008138 88467F         <1>         mov [esi+LD_CDirLevel], al
0000813B 89F7           <1>       mov    edi, esi
0000813D 81C780000000   <1>       add    edi, LD_CurrentDirectory
00008143 B920000000     <1>       mov    ecx, 32
00008148 F3AB           <1>       rep    stosd
                        <1>
                        <1> loc_ccdrv_get_FAT_volume_name_0:
0000814A 8A4603         <1>       mov    al, [esi+LD_FATType]
0000814D 08C0           <1>       or     al, al
0000814F 742A           <1>       jz     short loc_change_current_drv2
                        <1>
00008151 56             <1>       push   esi
00008152 3C02           <1>       cmp    al, 2
00008154 7705           <1>       ja     short loc_ccdrv_get_FAT32_vol_name
                        <1>
                        <1> loc_ccdrv_get_FAT2_16_vol_name:
00008156 83C631         <1>       add    esi, LD_BPB + VolumeLabel
00008159 EB03           <1>       jmp    short loc_ccdrv_get_FAT_volume_name_1
                        <1>
                        <1> loc_ccdrv_get_FAT32_vol_name:
0000815B 83C64D         <1>       add    esi, LD_BPB + FAT32_VolLab
                        <1> loc_ccdrv_get_FAT_volume_name_1:
0000815E 53             <1>       push   ebx
0000815F 56             <1>       push   esi
00008160 E8C3FEFFFF     <1>       call   get_FAT_volume_name
00008165 5F             <1>       pop    edi
00008166 5B             <1>       pop    ebx
                        <1>       ; BL = 0
00008167 720B           <1>       jc     short loc_change_current_drv1
00008169 20C0           <1>       and    al, al
0000816B 7407           <1>       jz     short loc_change_current_drv1
                        <1>
                        <1> loc_ccdrv_move_FAT_volume_name:
0000816D B90B000000     <1>       mov    ecx, 11
00008172 F3A4           <1>       rep    movsb
                        <1>
                        <1> loc_change_current_drv1:
00008174 5E             <1>       pop    esi
00008175 EB04           <1>       jmp    short loc_change_current_drv2
                        <1>
                        <1> loc_gmcs_init_drv_hd:
00008177 08E4           <1>       or     ah, ah
00008179 7404           <1>       jz     short loc_change_current_drv3
                        <1>       ; BL = 0, BH = Logical DOS drive number
                        <1> loc_change_current_drv2:
0000817B C6467E00       <1>       mov    byte [esi+LD_MediaChanged], 0
                        <1> loc_change_current_drv3:
0000817F 883D[56820100] <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>
00008185 8A4603         <1>       mov    al, [esi+LD_FATType]
00008188 A2[55820100]   <1>       mov    [Current_FATType], al
                        <1>
0000818D 8A26           <1>       mov    ah, [esi+LD_Name]
```

```
0000818F 8825[57820100]      <1>       mov    [Current_Dir_Drv], ah
                             <1>
00008195 20C0                <1>       and    al, al
00008197 741D                <1>       jz     short loc_restore_FS_current_directory
                             <1>
                             <1> loc_restore_FAT_current_directory:
00008199 8A667F              <1>       mov    ah, [esi+LD_CDirLevel]
0000819C 8825[54820100]      <1>       mov    [Current_Dir_Level], ah
000081A2 08E4                <1>       or     ah, ah
000081A4 7416                <1>         jz   short loc_ccdrv_reset_cdir_FAT_12_16_32_fcluster
                             <1>
000081A6 0FB6D4              <1>       movzx  edx, ah
000081A9 C0E204              <1>       shl    dl, 4 ; * 16
000081AC 01F2                <1>         add  edx, esi
000081AE 8B828C000000        <1>       mov    eax, [edx+LD_CurrentDirectory+12]
000081B4 EB2C                <1>       jmp    short loc_ccdrv_reset_cdir_FAT_fcluster
                             <1>
                             <1> loc_restore_FS_current_directory:
000081B6 E8BC4D0000          <1>       call   load_current_FS_directory
000081BB C3                  <1>       retn
                             <1>
                             <1> loc_ccdrv_reset_cdir_FAT_12_16_32_fcluster:
000081BC 3C03                <1>       cmp    al, 3
000081BE 7205                <1>       jb     short loc_ccdrv_reset_cdir_FAT_12_16_fcluster
                             <1> loc_ccdrv_reset_cdir_FAT32_fcluster:
000081C0 8B4632              <1>       mov    eax, [esi+LD_BPB+FAT32_RootFClust]
000081C3 EB04                <1>       jmp    short loc_ccdrv_check_rootdir_sign
                             <1> loc_ccdrv_reset_cdir_FAT_12_16_fcluster:
000081C5 30C0                <1>       xor    al, al  ; xor eax, eax
000081C7 31D2                <1>       xor    edx, edx
                             <1> loc_ccdrv_check_rootdir_sign:
000081C9 80BE8000000000      <1>       cmp    byte [esi+LD_CurrentDirectory], 0
000081D0 7510                <1>       jne    short loc_ccdrv_reset_cdir_FAT_fcluster
                             <1> loc_ccdrv_set_rootdir_FAT_fcluster:
000081D2 89868C000000        <1>         mov     [esi+LD_CurrentDirectory+12], eax
000081D8 C78680000000524F4F- <1>       mov    dword [esi+LD_CurrentDirectory], 'ROOT'
000081E1 54                  <1>
                             <1>
                             <1> loc_ccdrv_reset_cdir_FAT_fcluster:
000081E2 A3[50820100]        <1>       mov    [Current_Dir_FCluster], eax
                             <1>
000081E7 BF[B3890100]        <1>       mov    edi, PATH_Array
000081EC 89F2                <1>       mov    edx, esi
000081EE 81C680000000        <1>       add    esi, LD_CurrentDirectory
000081F4 B920000000          <1>       mov    ecx, 32
000081F9 F3A5                <1>       rep    movsd
                             <1>
000081FB E84C2D0000          <1>       call   change_prompt_dir_string
                             <1>
00008200 89D6                <1>       mov    esi, edx
                             <1>
00008202 29C0                <1>         sub  eax, eax
                             <1>        ;sub  edx, edx
00008204 BF[57820100]        <1>       mov    edi, Current_Dir_Drv
                             <1>
00008209 A2[23380100]        <1>       mov    [Restore_CDIR], al ; 0
0000820E 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
0000820F C705[108E0100]-     <1>       mov    dword [mainprog_return_addr], return_from_cmd_interpreter
00008215 [C3820000]          <1>
                             <1>
                             <1> loc_TRDOS_prompt:
00008219 BF[56830100]        <1>       mov    edi, TextBuffer
0000821E C6075B              <1>       mov    byte [edi], "["
00008221 47                  <1>       inc    edi
00008222 BE[76380100]        <1>       mov    esi, TRDOSPromptLabel
                             <1> get_next_prompt_label_char:
00008227 803E20              <1>       cmp    byte [esi], 20h
0000822A 7203                <1>       jb     short pass_prompt_label
0000822C A4                  <1>       movsb
0000822D EBF8                <1>       jmp    short get_next_prompt_label_char
                             <1> pass_prompt_label:
0000822F C6075D              <1>       mov    byte [edi], "]"
00008232 47                  <1>       inc    edi
00008233 C60720              <1>       mov    byte [edi], 20h
00008236 47                  <1>       inc    edi
00008237 BE[57820100]        <1>       mov    esi, Current_Dir_Drv
0000823C 66A5                <1>       movsw
0000823E A4                  <1>       movsb
                             <1> loc_prompt_current_directory:
0000823F 803E20              <1>       cmp    byte [esi], 20h
00008242 7203                <1>       jb     short pass_prompt_current_directory
00008244 A4                  <1>       movsb
00008245 EBF8                <1>       jmp    short loc_prompt_current_directory
                             <1> pass_prompt_current_directory:
00008247 C6073E              <1>       mov    byte [edi], '>'
0000824A 47                  <1>       inc    edi
0000824B C60700              <1>       mov    byte [edi], 0
0000824E BE[56830100]        <1>       mov    esi, TextBuffer
00008253 E8EDF2FFFF          <1>       call   print_msg
                             <1>
                             <1>       ;sub   bh, bh ; video page = 0
                             <1>       ;call  get_cpos ; get cursor position
00008258 668B15[AE810100]    <1>       mov    dx, [CURSOR_POSN] ; video page 0
0000825F 8815[B6820100]      <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:
00008265 E899000000     <1>       call   rw_char
                        <1> loc_move_command:
0000826A BE[06830100]   <1>       mov    esi, CommandBuffer
0000826F 89F7           <1>       mov    edi, esi
00008271 31C9           <1>       xor    ecx, ecx
                        <1> first_command_char:
00008273 AC             <1>       lodsb
00008274 3C20           <1>       cmp    al, 20h
00008276 772E           <1>       ja     short pass_space_control
00008278 7241           <1>       jb     short loc_move_cmd_arguments_ok
0000827A 81FE[55830100] <1>       cmp    esi, CommandBuffer + 79
00008280 72F1           <1>       jb     short first_command_char
00008282 EB37           <1>       jmp    short loc_move_cmd_arguments_ok
                        <1>
                        <1> next_command_char:
00008284 AC             <1>       lodsb
00008285 3C20           <1>       cmp    al, 20h
00008287 771D           <1>       ja     short pass_space_control
00008289 7230           <1>       jb     short loc_move_cmd_arguments_ok
                        <1>
                        <1> loc_1st_cmd_arg: ; 30/01/2016
0000828B AC             <1>       lodsb
0000828C 3C20           <1>       cmp    al, 20h
0000828E 74FB           <1>       je     short loc_1st_cmd_arg
00008290 7229           <1>       jb     short loc_move_cmd_arguments_ok
                        <1>
00008292 C60700         <1>         mov     byte [edi], 0
00008295 47             <1>       inc    edi
                        <1>
                        <1> loc_move_cmd_arguments:
00008296 AA             <1>       stosb
00008297 81FE[55830100] <1>       cmp    esi, CommandBuffer + 79
0000829D 731C           <1>       jnb    short loc_move_cmd_arguments_ok
0000829F AC             <1>         lodsb
000082A0 3C20           <1>       cmp    al, 20h
000082A2 73F2           <1>       jnb    short loc_move_cmd_arguments
000082A4 EB15           <1>       jmp    short loc_move_cmd_arguments_ok
                        <1>
                        <1> pass_space_control:
000082A6 3C61           <1>       cmp    al, 61h
000082A8 7206           <1>       jb     short pass_capitalize
000082AA 3C7A           <1>       cmp    al, 7Ah
000082AC 7702           <1>       ja     short pass_capitalize
000082AE 24DF           <1>       and    al, 0DFh
                        <1> pass_capitalize:
000082B0 AA             <1>       stosb
000082B1 FEC1           <1>       inc     cl
000082B3 81FE[55830100] <1>         cmp     esi, CommandBuffer + 79
000082B9 72C9           <1>       jb      short next_command_char
                        <1>
                        <1> loc_move_cmd_arguments_ok:
000082BB C60700         <1>         mov     byte [edi], 0
                        <1>
                        <1> call_command_interpreter:
000082BE E8CF080000     <1>       call    command_interpreter
                        <1>
                        <1> return_from_cmd_interpreter:
000082C3 B950000000     <1>         mov  ecx, 80
                        <1>       ;mov   cx, 80
000082C8 BF[06830100]   <1>       mov    edi, CommandBuffer
000082CD 30C0           <1>       xor    al, al
000082CF F3AA           <1>       rep    stosb
                        <1>       ;cmp   byte [Program_Exit], 0
                        <1>       ;ja    short loc_terminate_trdos
                        <1>
                        <1>       ; 16/01/2016
000082D1 803D[BA6F0000]03 <1>     cmp    byte [CRT_MODE], 3 ; 80*25 color
000082D8 741D           <1>       je     short pass_set_txt_mode
                        <1>
000082DA E88798FFFF     <1>       call   set_txt_mode ; set vide mode to 03h
                        <1>       ; 07/01/2017
000082DF 30C0           <1>       xor    al, al
                        <1>
                        <1> loc_check_active_page:
                        <1>       ;xor   al, al
000082E1 3805[BE810100] <1>       cmp    [ACTIVE_PAGE], al ; 0
000082E7 0F842CFFFFFF   <1>         je      loc_TRDOS_prompt
                        <1>       ; AL = 0 = video page 0
000082ED E8BD9CFFFF     <1>       call   set_active_page
000082F2 E922FFFFFF     <1>         jmp     loc_TRDOS_prompt ; infinitive loop
                        <1>
                        <1> pass_set_txt_mode:
000082F7 BE[BF440100]   <1>       mov    esi, nextline
000082FC E844F2FFFF     <1>       call   print_msg
00008301 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:
00008303 30E4           <1>       xor     ah, ah
00008305 E8DB8BFFFF     <1>       call   int16h
0000830A 20C0           <1>       and    al, al
0000830C 7432           <1>       jz     short loc_arrow
0000830E 3CE0           <1>       cmp    al, 0E0h
00008310 742E           <1>       je     short loc_arrow
00008312 3C08           <1>       cmp    al, 08h
00008314 7542           <1>       jne    short char_return
                        <1> loc_back:
00008316 3A15[B6820100] <1>       cmp    dl, [CursorColumn]
0000831C 76E5           <1>       jna     short readnextchar
                        <1> prev_column:
0000831E FECA           <1>       dec    dl
                        <1> set_cursor_pos:
                        <1>       ;push  dx
00008320 52             <1>       push   edx ; 29/12/2017
                        <1>       ;xor   bh, bh ; 0 = video page 0
                        <1>       ; DH = Row, DL = Column
00008321 E871A0FFFF     <1>       call   _set_cpos ; 17/01/2016
00008326 5A             <1>         pop  edx ; 29/12/2017
                        <1>       ;pop   dx
                        <1>       ;movzx ebx, dl
00008327 88D3           <1>       mov    bl, dl
00008329 2A1D[B6820100] <1>       sub    bl, [CursorColumn]
0000832F B020           <1>       mov    al, 20h
00008331 8883[06830100] <1>       mov    [CommandBuffer+ebx], al
                        <1>       ;sub   bh, bh ; video page 0
                        <1>       ;mov   cx, 1
00008337 B307           <1>       mov    bl, 7 ; color attribute
00008339 E8389FFFFF     <1>       call   _write_c_current ; 17/01/2016
                        <1>       ;mov   dx, [CURSOR_POSN]
0000833E EBC3           <1>       jmp    short readnextchar
                        <1> loc_arrow:
00008340 80FC4B         <1>       cmp    ah, 4Bh
00008343 74D1           <1>       je     short loc_back
00008345 80FC53         <1>       cmp    ah, 53h
00008348 74CC           <1>       je      short loc_back
0000834A 80FC4D         <1>       cmp    ah, 4Dh
0000834D 75B4           <1>       jne    short readnextchar
0000834F 80FA4F         <1>       cmp    dl, 79
00008352 73AF           <1>       jnb    short readnextchar
00008354 FEC2           <1>       inc    dl
00008356 EBC8           <1>       jmp    short set_cursor_pos
                        <1> char_return:
00008358 0FB6DA         <1>       movzx  ebx, dl
0000835B 2A1D[B6820100] <1>       sub    bl, [CursorColumn]
00008361 3C20           <1>       cmp    al, 20h
00008363 721D           <1>       jb     short loc_escape
00008365 8883[06830100] <1>       mov    [CommandBuffer+ebx], al
0000836B 80FA4F         <1>       cmp    dl, 79
0000836E 7393           <1>       jnb    short readnextchar
00008370 66BB0700       <1>       mov    bx, 7 ; color attribute
00008374 E8849FFFFF     <1>       call   _write_tty
00008379 668B15[AE810100] <1>     mov    dx, [CURSOR_POSN] ; video page 0
00008380 EB81           <1>         jmp     readnextchar
                        <1> loc_escape:
00008382 3C1B           <1>       cmp    al, 1Bh
00008384 7418           <1>       je     short rw_char_retn
                        <1>       ;
00008386 3C0D           <1>       cmp    al, 0Dh ; CR
00008388 0F8575FFFFFF   <1>         jne     readnextchar
                        <1>       ; 13/05/2016
0000838E 66BB0700       <1>       mov    bx, 7 ; attribute/color (bl)
                        <1>                    ; video page 0 (bh=0)
00008392 E8669FFFFF     <1>       call   _write_tty
                        <1>       ;mov   bx, 7  ; attribute/color
                        <1>                    ; video page 0 (bh=0)
00008397 B00A           <1>       mov    al, 0Ah ; LF
00008399 E85F9FFFFF     <1>       call   _write_tty
                        <1> rw_char_retn:
0000839E 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
0000839F E86DE7FFFF     <1>       call   RTC_40 ; GET RTC DATE
                        <1>
000083A4 88D0           <1>       mov    al, dl
000083A6 E82D8BFFFF     <1>       call   bcd_to_ascii
000083AB 66A3[62390100] <1>       mov    [Day], ax
                        <1>
000083B1 88F0           <1>       mov    al, dh
000083B3 E8208BFFFF     <1>       call   bcd_to_ascii
000083B8 66A3[65390100] <1>       mov    [Month], ax
                        <1>
000083BE 88E8           <1>       mov    al, ch
000083C0 E8138BFFFF     <1>       call   bcd_to_ascii
000083C5 66A3[68390100] <1>       mov    [Century], ax
                        <1>
000083CB 88C8           <1>       mov    al, cl
```

```
000083CD E8068BFFFF          <1>       call   bcd_to_ascii
000083D2 66A3[6A390100]      <1>       mov    word [Year], ax
                             <1>
000083D8 BE[52390100]        <1>       mov    esi, Msg_Show_Date
000083DD E863F1FFFF          <1>       call   print_msg
                             <1>
000083E2 C3                  <1>       retn
                             <1>
                             <1> set_date:
                             <1>       ; 13/05/2016
                             <1>       ; 18/01/2016 (TRDOS 386 = TRDOS v2.0)
                             <1>         ; 2004-2005
                             <1>
000083E3 BE[36390100]        <1>       mov    esi, Msg_Enter_Date
000083E8 E858F1FFFF          <1>       call   print_msg
                             <1>
                             <1> loc_enter_day_1:
000083ED 30E4                <1>       xor     ah, ah
000083EF E8F18AFFFF          <1>       call   int16h
                             <1>       ; AL = ASCII Code of the Character
000083F4 3C0D                <1>       cmp    al, 13
000083F6 0F84B7010000        <1>       je     loc_set_date_retn
000083FC 3C1B                <1>       cmp    al, 27
000083FE 0F84AF010000        <1>       je     loc_set_date_retn
00008404 A2[62390100]        <1>       mov    [Day], al
00008409 3C30                <1>       cmp    al, '0'
0000840B 0F82AD010000        <1>       jb     loc_set_date_stc_0
00008411 3C33                <1>       cmp    al, '3'
00008413 0F87A5010000        <1>       ja     loc_set_date_stc_0
                             <1>       ; 13/05/2016
                             <1>       ;mov   bx, 7 ; attribute/color (bl)
                             <1>                    ; video page 0 (bh)
00008419 B307                <1>       mov    bl, 7
0000841B E8DD9EFFFF          <1>       call   _write_tty
                             <1> loc_enter_day_2:
00008420 30E4                <1>       xor     ah, ah
00008422 E8BE8AFFFF          <1>       call   int16h
                             <1>       ; AL = ASCII Code of the Character
00008427 3C1B                <1>       cmp    al, 27
00008429 0F8484010000        <1>         je      loc_set_date_retn
0000842F A2[63390100]        <1>       mov    [Day+1], al
00008434 3C30                <1>       cmp    al, '0'
00008436 0F828C010000        <1>         jb      loc_set_date_stc_1
0000843C 3C39                <1>       cmp    al, '9'
0000843E 0F8784010000        <1>         ja      loc_set_date_stc_1
00008444 803D[62390100]33    <1>       cmp    byte [Day], '3'
0000844B 7208                <1>       jb     short pass_set_day_31
0000844D 3C31                <1>       cmp    al, '1'
0000844F 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)
00008455 B307                <1>       mov    bl, 7
00008457 E8A19EFFFF          <1>       call   _write_tty
                             <1> loc_enter_separator_1:
0000845C 28E4                <1>       sub     ah, ah ; 0
0000845E E8828AFFFF          <1>       call   int16h
                             <1>       ; AL = ASCII Code of the Character
00008463 3C1B                <1>       cmp    al, 27
00008465 0F8448010000        <1>         je      loc_set_date_retn
0000846B 3C2D                <1>       cmp    al, '-'
0000846D 7408                <1>       je     short pass_set_date_separator_1
0000846F 3C2F                <1>       cmp    al, '/'
00008471 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)
00008477 B307                <1>       mov    bl, 7
00008479 E87F9EFFFF          <1>       call   _write_tty
                             <1> loc_enter_month_1:
0000847E 30E4                <1>       xor     ah, ah ; 0
00008480 E8608AFFFF          <1>       call   int16h
                             <1>       ; AL = ASCII Code of the Character
00008485 3C1B                <1>       cmp    al, 27
00008487 0F8426010000        <1>         je      loc_set_date_retn
0000848D A2[65390100]        <1>       mov    [Month], al
00008492 3C30                <1>       cmp    al, '0'
00008494 0F8264010000        <1>         jb      loc_set_date_stc_3
0000849A 3C31                <1>       cmp    al, '1'
0000849C 0F875C010000        <1>         ja      loc_set_date_stc_3
                             <1>       ; 13/05/2016
                             <1>       ;mov   bx, 7 ; attribute/color (bl)
                             <1>                    ; video page 0 (bh)
000084A2 B307                <1>       mov    bl, 7
000084A4 E8549EFFFF          <1>       call   _write_tty
                             <1> loc_enter_month_2:
000084A9 30E4                <1>       xor     ah, ah
000084AB E8358AFFFF          <1>       call   int16h
                             <1>       ; AL = ASCII Code of the Character
000084B0 3C1B                <1>       cmp    al, 27
000084B2 0F84FB000000        <1>         je      loc_set_date_retn
000084B8 A2[66390100]        <1>       mov    [Month+1], al
000084BD 3C30                <1>       cmp    al, '0'
000084BF 0F8254010000        <1>         jb      loc_set_date_stc_4
000084C5 3C39                <1>       cmp    al, '9'
000084C7 0F874C010000        <1>         ja      loc_set_date_stc_4
000084CD 803D[65390100]31    <1>       cmp    byte [Month], '1'
000084D4 7208                <1>       jb     short pass_set_month_12
000084D6 3C32                <1>       cmp    al, '2'
000084D8 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)
000084DE B307           <1>       mov   bl, 7
000084E0 E8189EFFFF     <1>       call  _write_tty
                        <1> loc_enter_separator_2:
000084E5 28E4           <1>       sub    ah, ah
000084E7 E8F989FFFF     <1>       call  int16h
                        <1>       ; AL = ASCII Code of the Character
000084EC 3C1B           <1>       cmp   al, 27
000084EE 0F84BF000000   <1>         je     loc_set_date_retn
000084F4 3C2D           <1>       cmp   al, '-'
000084F6 7408           <1>       je    short pass_set_date_separator_2
000084F8 3C2F           <1>       cmp   al, '/'
000084FA 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)
00008500 B307           <1>       mov   bl, 7
00008502 E8F69DFFFF     <1>       call  _write_tty
                        <1> loc_enter_year_1:
00008507 30E4           <1>       xor    ah, ah
00008509 E8D789FFFF     <1>       call  int16h
                        <1>       ; AL = ASCII Code of the Character
0000850E 3C1B           <1>       cmp   al, 27
00008510 0F849D000000   <1>         je     loc_set_date_retn
00008516 A2[6A390100]   <1>       mov   [Year], al
0000851B 3C30           <1>       cmp   al, '0'
0000851D 0F822C010000   <1>         jb     loc_set_date_stc_6
00008523 3C39           <1>       cmp   al, '9'
00008525 0F8724010000   <1>         ja     loc_set_date_stc_6
                        <1>       ; 13/05/2016
                        <1>       ;mov   bx, 7 ; attribute/color (bl)
                        <1>                     ; video page 0 (bh)
0000852B B307           <1>       mov   bl, 7
0000852D E8CB9DFFFF     <1>       call  _write_tty
                        <1> loc_enter_year_2:
00008532 30E4           <1>       xor    ah, ah
00008534 E8AC89FFFF     <1>       call  int16h
                        <1>       ; AL = ASCII Code of the Character
00008539 3C1B           <1>       cmp   al, 27
0000853B 7476           <1>       je    short loc_set_date_retn
0000853D A2[6B390100]   <1>       mov   byte [Year+1], al
00008542 3C30           <1>       cmp   al, '0'
00008544 0F8220010000   <1>         jb     loc_set_date_stc_7
0000854A 3C39           <1>       cmp   al, '9'
0000854C 0F8718010000   <1>         ja     loc_set_date_stc_7
                        <1>       ; 13/05/2016
                        <1>       ;mov   bx, 7 ; attribute/color (bl)
                        <1>                     ; video page 0 (bh)
00008552 B307           <1>       mov   bl, 7
00008554 E8A49DFFFF     <1>       call  _write_tty
                        <1> loc_set_date_get_lchar_again:
00008559 28E4           <1>       sub   ah, ah ; 0
0000855B E88589FFFF     <1>       call  int16h
                        <1>       ; AL = ASCII Code of the Character
00008560 3C0D           <1>       cmp   al, 13 ; ENTER key
00008562 7412           <1>       je    short loc_set_date_progress
00008564 3C1B           <1>       cmp   al, 27 ; ESC key
00008566 744B           <1>       je    short loc_set_date_retn
                        <1>       ;
00008568 E82A010000     <1>       call  check_for_backspace
0000856D 75EA           <1>       jne   short loc_set_date_get_lchar_again
                        <1>
                        <1> loc_set_date_bs_8:
0000856F E811010000     <1>       call  write_backspace
00008574 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
00008576 E896E5FFFF     <1>       call  RTC_40 ; GET RTC DATE
                        <1>       ; CH = century (in BCD)
                        <1>
0000857B 66A1[6A390100] <1>       mov   ax, [Year]
00008581 662D3030       <1>       sub   ax, '00'
00008585 C0E004         <1>       shl   al, 4 ; * 16
00008588 88C1           <1>       mov   cl, al
0000858A 00E1           <1>       add   cl, ah
0000858C 66A1[65390100] <1>       mov   ax, [Month]
00008592 662D3030       <1>       sub   ax, '00'
00008596 C0E004         <1>       shl   al, 4 ; * 16
00008599 88C6           <1>       mov   dh, al
0000859B 00E6           <1>       add   dh, ah
0000859D 66A1[62390100] <1>       mov   ax, [Day]
000085A3 662D3030       <1>       sub   ax, '00'
000085A7 C0E004         <1>       shl   al, 4 ; * 16
000085AA 88C2           <1>       mov   dl, al
000085AC 00E2           <1>       add   dl, ah
                        <1>
                        <1>       ;mov   ah, 05h
                        <1>       ;call  int1Ah
000085AE E88BE5FFFF     <1>       call  RTC_50 ; SET RTC DATE
                        <1>
                        <1> loc_set_date_retn:
000085B3 BE[BF440100]   <1>       mov   esi, nextline
000085B8 E888EFFFFF     <1>       call  print_msg
000085BD C3             <1>       retn
                        <1>
                        <1> loc_set_date_stc_0:
                        <1>       ;xor   bh, bh ; video page 0
000085BE E81E9EFFFF     <1>       call  beeper ; BEEP !
000085C3 E925FEFFFF     <1>         jmp     loc_enter_day_1
                        <1> loc_set_date_stc_1:
000085C8 E8CA000000     <1>       call  check_for_backspace
```

```
000085CD 740A              <1>       je     short loc_set_date_bs_1
                           <1>       ;xor   bh, bh ; video page 0
000085CF E80D9EFFFF        <1>       call   beeper ; BEEP !
000085D4 E947FEFFFF        <1>         jmp     loc_enter_day_2
                           <1> loc_set_date_bs_1:
000085D9 E8A7000000        <1>       call   write_backspace
000085DE E90AFEFFFF        <1>         jmp     loc_enter_day_1
                           <1> loc_set_date_stc_2:
000085E3 E8AF000000        <1>       call   check_for_backspace
000085E8 740A              <1>       je     short loc_set_date_bs_2
                           <1>       ;xor   bh, bh ; video page 0
000085EA E8F29DFFFF        <1>       call   beeper ; BEEP !
000085EF E968FEFFFF        <1>         jmp     loc_enter_separator_1
                           <1> loc_set_date_bs_2:
000085F4 E88C000000        <1>       call   write_backspace
000085F9 E922FEFFFF        <1>         jmp     loc_enter_day_2
                           <1> loc_set_date_stc_3:
000085FE E894000000        <1>       call   check_for_backspace
00008603 740A              <1>       je short loc_set_date_bs_3
                           <1>       ;xor   bh, bh ; video page 0
00008605 E8D79DFFFF        <1>       call   beeper ; BEEP !
0000860A E96FFEFFFF        <1>         jmp     loc_enter_month_1
                           <1> loc_set_date_bs_3:
0000860F E871000000        <1>       call   write_backspace
00008614 E943FEFFFF        <1>         jmp     loc_enter_separator_1
                           <1> loc_set_date_stc_4:
00008619 E879000000        <1>       call   check_for_backspace
0000861E 740A              <1>       je     short loc_set_date_bs_4
                           <1>       ;xor   bh, bh ; video page 0
00008620 E8BC9DFFFF        <1>       call   beeper ; BEEP !
00008625 E97FFEFFFF        <1>         jmp     loc_enter_month_2
                           <1> loc_set_date_bs_4:
0000862A E856000000        <1>       call   write_backspace
0000862F E94AFEFFFF        <1>         jmp     loc_enter_month_1
                           <1> loc_set_date_stc_5:
00008634 E85E000000        <1>       call   check_for_backspace
00008639 740A              <1>       je     short loc_set_date_bs_5
                           <1>       ;xor   bh, bh ; video page 0
0000863B E8A19DFFFF        <1>       call   beeper ; BEEP !
00008640 E9A0FEFFFF        <1>         jmp     loc_enter_separator_2
                           <1> loc_set_date_bs_5:
00008645 E83B000000        <1>       call   write_backspace
0000864A E95AFEFFFF        <1>         jmp     loc_enter_month_2
                           <1> loc_set_date_stc_6:
0000864F E843000000        <1>       call   check_for_backspace
00008654 740A              <1>         je      short  loc_set_date_bs_6
                           <1>       ;xor   bh, bh ; video page 0
00008656 E8869DFFFF        <1>       call   beeper ; BEEP !
0000865B E9A7FEFFFF        <1>         jmp     loc_enter_year_1
                           <1> loc_set_date_bs_6:
00008660 E820000000        <1>       call   write_backspace
00008665 E97BFEFFFF        <1>         jmp     loc_enter_separator_2
                           <1> loc_set_date_stc_7:
0000866A E828000000        <1>       call   check_for_backspace
0000866F 740A              <1>       je     short loc_set_date_bs_7
                           <1>       ;xor   bh, bh ; video page 0
00008671 E86B9DFFFF        <1>       call   beeper ; BEEP !
00008676 E9B7FEFFFF        <1>         jmp     loc_enter_year_2
                           <1> loc_set_date_bs_7:
0000867B E805000000        <1>       call   write_backspace
00008680 E982FEFFFF        <1>         jmp     loc_enter_year_1
                           <1>
                           <1> write_backspace:
                           <1>       ; 18/01/2016 (TRDOS 386 = TRDOS v2.0)
00008685 B008              <1>       mov    al, 08h ; BACKSPACE
                           <1>       ; 13/05/2016
00008687 66BB0700          <1>       mov    bx, 7 ; bl = attribute/color
                           <1>                    ; bh = video page = 0
0000868B E86D9CFFFF        <1>       call   _write_tty
00008690 B020              <1>       mov    al, 20h ; BLANK/SPACE char
                           <1>       ;mov   bx, 7 ; attribute/color
                           <1>       ;call   write_c_current
                           <1>       ;retn
00008692 E9DF9BFFFF        <1>       jmp    _write_c_current
                           <1>
                           <1> check_for_backspace:
                           <1>       ; 18/01/2016 (TRDOS 386 = TRDOS v2.0)
00008697 663D080E          <1>       cmp    ax, 0E08h
0000869B 7410              <1>       je     short cfbs_retn
0000869D 663DE04B          <1>       cmp    ax, 4BE0h
000086A1 740A              <1>       je     short cfbs_retn
000086A3 663D004B          <1>       cmp    ax, 4B00h
000086A7 7404              <1>       je     short cfbs_retn
000086A9 663DE053          <1>       cmp    ax, 53E0h
                           <1> cfbs_retn:
000086AD 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
000086AE E8EDE3FFFF        <1>       call   RTC_20 ; GET RTC TIME
                           <1>
000086B3 88E8              <1>       mov    al, ch
000086B5 E81E88FFFF        <1>       call   bcd_to_ascii
000086BA 66A3[90390100]    <1>       mov    [Hour], ax
                           <1>
000086C0 88C8              <1>       mov    al, cl
000086C2 E81188FFFF        <1>       call   bcd_to_ascii
000086C7 66A3[93390100]    <1>       mov    [Minute], ax
                           <1>
000086CD 88F0              <1>       mov    al, dh
000086CF E80488FFFF        <1>       call   bcd_to_ascii
```

```
803 000086D4 66A3[96390100]      <1>       mov    [Second], ax
804                              <1>
805 000086DA BE[80390100]        <1>       mov    esi, Msg_Show_Time
806 000086DF E861EEFFFF          <1>       call   print_msg
807 000086E4 C3                  <1>       retn
808                              <1>
809                              <1> set_time:
810                              <1>       ; 13/05/2016
811                              <1>       ; 18/01/2016 (TRDOS 386 = TRDOS v2.0)
812                              <1>         ; 2004-2005
813                              <1>
814 000086E5 BE[6F390100]        <1>       mov    esi, Msg_Enter_Time
815 000086EA E856EEFFFF          <1>       call   print_msg
816                              <1>
817                              <1> loc_enter_hour_1:
818 000086EF 30E4                <1>       xor     ah, ah
819 000086F1 E8EF87FFFF          <1>       call   int16h
820                              <1>       ; AL = ASCII Code of the Character
821 000086F6 3C0D                <1>       cmp    al, 13 ; ENTER key
822 000086F8 0F84AE010000        <1>         je      loc_set_time_retn
823 000086FE 3C1B                <1>       cmp    al, 27 ; ESC key
824 00008700 0F84A6010000        <1>         je      loc_set_time_retn
825 00008706 A2[90390100]        <1>       mov    [Hour], al
826 0000870B 3C30                <1>       cmp    al, '0'
827 0000870D 0F82A4010000        <1>         jb      loc_set_time_stc_0
828 00008713 3C32                <1>       cmp    al, '2'
829 00008715 0F879C010000        <1>         ja      loc_set_time_stc_0
830                              <1>       ; 13/05/2016
831                              <1>       ;mov   bx, 7 ; attribute/color (bl)
832                              <1>                    ; video page 0 (bh)
833 0000871B B307                <1>       mov    bl, 7
834 0000871D E8DB9BFFFF          <1>       call   _write_tty
835                              <1> loc_enter_hour_2:
836 00008722 30E4                <1>       xor     ah, ah
837 00008724 E8BC87FFFF          <1>       call   int16h
838                              <1>       ; AL = ASCII Code of the Character
839 00008729 3C1B                <1>       cmp    al, 27
840 0000872B 0F847B010000        <1>         je      loc_set_time_retn
841 00008731 A2[91390100]        <1>       mov    [Hour+1], al
842 00008736 3C30                <1>       cmp    al, '0'
843 00008738 0F8283010000        <1>         jb      loc_set_time_stc_1
844 0000873E 3C39                <1>       cmp    al, '9'
845 00008740 0F877B010000        <1>       ja     loc_set_time_stc_1
846 00008746 803D[90390100]32    <1>         cmp     byte [Hour], '2'
847 0000874D 7208                <1>       jb     short pass_set_time_24
848 0000874F 3C34                <1>       cmp    al, '4'
849 00008751 0F876A010000        <1>         ja      loc_set_time_stc_1
850                              <1> pass_set_time_24:
851                              <1>       ; 13/05/2016
852                              <1>       ;mov   bx, 7 ; attribute/color (bl)
853                              <1>                    ; video page 0 (bh)
854 00008757 B307                <1>       mov    bl, 7
855 00008759 E89F9BFFFF          <1>       call   _write_tty
856                              <1> loc_enter_time_separator_1:
857 0000875E 28E4                <1>       sub    ah, ah ; 0
858 00008760 E88087FFFF          <1>       call   int16h
859                              <1>       ; AL = ASCII Code of the Character
860 00008765 3C1B                <1>       cmp    al, 27
861 00008767 0F843F010000        <1>         je      loc_set_time_retn
862 0000876D 3C3A                <1>       cmp    al, ':'
863 0000876F 0F8567010000        <1>         jne     loc_set_time_stc_2
864                              <1>       ; 13/05/2016
865                              <1>       ;mov   bx, 7 ; attribute/color (bl)
866                              <1>                    ; video page 0 (bh)
867 00008775 B307                <1>       mov    bl, 7
868 00008777 E8819BFFFF          <1>       call   _write_tty
869                              <1> loc_enter_minute_1:
870 0000877C 30E4                <1>       xor     ah, ah
871 0000877E E86287FFFF          <1>       call   int16h
872                              <1>       ; AL = ASCII Code of the Character
873 00008783 3C1B                <1>       cmp    al, 27
874 00008785 0F8421010000        <1>         je      loc_set_time_retn
875 0000878B A2[93390100]        <1>       mov    [Minute], al
876 00008790 3C30                <1>       cmp    al, '0'
877 00008792 0F825F010000        <1>         jb      loc_set_time_stc_3
878 00008798 3C35                <1>       cmp    al, '5'
879 0000879A 0F8757010000        <1>         ja      loc_set_time_stc_3
880                              <1>       ; 13/05/2016
881                              <1>       ;mov   bx, 7 ; attribute/color (bl)
882                              <1>                    ; video page 0 (bh)
883 000087A0 B307                <1>       mov    bl, 7
884 000087A2 E8569BFFFF          <1>       call   _write_tty
885                              <1> loc_enter_minute_2:
886 000087A7 30E4                <1>       xor     ah, ah
887 000087A9 E83787FFFF          <1>       call   int16h
888                              <1>       ; AL = ASCII Code of the Character
889 000087AE 3C1B                <1>       cmp    al, 27
890 000087B0 0F84F6000000        <1>         je      loc_set_time_retn
891 000087B6 A2[94390100]        <1>       mov    [Minute+1], al
892 000087BB 3C30                <1>       cmp    al, '0'
893 000087BD 0F824F010000        <1>         jb      loc_set_time_stc_4
894 000087C3 3C39                <1>       cmp    al, '9'
895 000087C5 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 000087CB B307                <1>       mov    bl, 7
900 000087CD E82B9BFFFF          <1>       call   _write_tty
901                              <1> loc_enter_time_separator_2:
902 000087D2 66C705[96390100]30- <1>       mov    word [Second], 3030h
902 000087DA 30                  <1>
903 000087DB 28E4                <1>       sub     ah, ah
904 000087DD E80387FFFF          <1>       call   int16h
905                              <1>       ; AL = ASCII Code of the Character
906 000087E2 3C0D                <1>       cmp    al, 13
```

```
000087E4 0F8485000000          <1>         je      loc_set_time_progress
000087EA 3C1B                  <1>       cmp    al, 27
000087EC 0F84BA000000          <1>         je      loc_set_time_retn
000087F2 3C3A                  <1>       cmp    al, ':'
000087F4 0F8533010000          <1>         jne     loc_set_time_stc_5
                               <1>       ; 13/05/2016
                               <1>       ;mov   bx, 7 ; attribute/color (bl)
                               <1>                    ; video page 0 (bh)
000087FA B307                  <1>       mov    bl, 7
000087FC E8FC9AFFFF            <1>       call   _write_tty
                               <1> loc_enter_second_1:
00008801 30E4                  <1>       xor     ah, ah
00008803 E8DD86FFFF            <1>       call   int16h
                               <1>       ; AL = ASCII Code of the Character
00008808 3C0D                  <1>       cmp    al, 13
0000880A 7463                  <1>       je     short loc_set_time_progress
0000880C 3C1B                  <1>       cmp    al, 27
0000880E 0F8498000000          <1>         je      loc_set_time_retn
00008814 A2[96390100]          <1>       mov    [Second], al
00008819 3C30                  <1>       cmp    al, '0'
0000881B 0F8227010000          <1>         jb      loc_set_time_stc_6
00008821 3C35                  <1>       cmp    al, '5'
00008823 0F871F010000          <1>         ja      loc_set_time_stc_6
                               <1>       ; 13/05/2016
                               <1>       ;mov   bx, 7 ; attribute/color (bl)
                               <1>                    ; video page 0 (bh)
00008829 B307                  <1>       mov    bl, 7
0000882B E8CD9AFFFF            <1>       call   _write_tty
                               <1> loc_enter_second_2:
00008830 30E4                  <1>       xor     ah, ah
00008832 E8AE86FFFF            <1>       call   int16h
                               <1>       ; AL = ASCII Code of the Character
00008837 3C1B                  <1>       cmp    al, 27
00008839 7471                  <1>       je     short loc_set_time_retn
0000883B 3C30                  <1>       cmp    al, '0'
0000883D 0F8229010000          <1>         jb      loc_set_time_stc_7
00008843 3C39                  <1>       cmp    al, '9'
00008845 0F8721010000          <1>         ja      loc_set_time_stc_7
                               <1>       ; 13/05/2016
                               <1>       ;mov   bx, 7 ; attribute/color (bl)
                               <1>                    ; video page 0 (bh)
0000884B B307                  <1>       mov    bl, 7
0000884D E8AB9AFFFF            <1>       call   _write_tty
                               <1> loc_set_time_get_lchar_again:
00008852 28E4                  <1>       sub    ah, ah ; 0
00008854 E88C86FFFF            <1>       call   int16h
                               <1>       ; AL = ASCII Code of the Character
00008859 3C0D                  <1>       cmp    al, 13
0000885B 7412                  <1>       je     short loc_set_time_progress
0000885D 3C1B                  <1>       cmp    al, 27
0000885F 744B                  <1>       je     short loc_set_time_retn
                               <1>       ;
00008861 E831FEFFFF            <1>       call   check_for_backspace
00008866 75EA                  <1>       jne    short loc_set_time_get_lchar_again
                               <1>
                               <1> loc_set_time_bs_8:
00008868 E818FEFFFF            <1>       call   write_backspace
0000886D EBC1                  <1>       jmp    short loc_enter_second_2
                               <1>
                               <1> loc_set_time_progress:
                               <1>       ; Get Current Time
                               <1>       ;mov   ah, 02h
                               <1>       ;call  int1Ah
0000886F E82CE2FFFF            <1>       call   RTC_20 ; GET RTC TIME
                               <1>       ;DL = Daylight Savings Enable option (0-1)
                               <1>
00008874 66A1[90390100]        <1>       mov    ax, [Hour]
0000887A 662D3030              <1>       sub    ax, '00'
0000887E C0E004                <1>       shl    al, 4 ; * 16
00008881 88C5                  <1>       mov    ch, al
00008883 00E5                  <1>       add    ch, ah
00008885 66A1[93390100]        <1>       mov    ax, [Minute]
0000888B 662D3030              <1>       sub    ax, '00'
0000888F C0E004                <1>       shl    al, 4 ; * 16
00008892 88C1                  <1>       mov    cl, al
00008894 00E1                  <1>       add    cl, ah
00008896 66A1[96390100]        <1>       mov    ax, [Second]
0000889C 662D3030              <1>       sub    ax, '00'
000088A0 C0E004                <1>       shl    al, 4 ; * 16
000088A3 88C6                  <1>       mov    dh, al
000088A5 00E6                  <1>       add    dh, ah
                               <1>
                               <1>       ;mov   ah, 03h
                               <1>       ;call  int1Ah
000088A7 E823E2FFFF            <1>       call   RTC_30 ; SET RTC TIME
                               <1>
                               <1> loc_set_time_retn:
000088AC BE[BF440100]          <1>       mov    esi, nextline
000088B1 E88FECFFFF            <1>       call   print_msg
000088B6 C3                    <1>       retn
                               <1>
                               <1> loc_set_time_stc_0:
                               <1>       ;xor   bh, bh ; video page 0
000088B7 E8259BFFFF            <1>       call   beeper ; BEEP !
000088BC E92EFEFFFF            <1>         jmp     loc_enter_hour_1
                               <1> loc_set_time_stc_1:
000088C1 E8D1FDFFFF            <1>       call   check_for_backspace
000088C6 740A                  <1>       je     short loc_set_time_bs_1
                               <1>       ;xor   bh, bh ; video page 0
000088C8 E8149BFFFF            <1>       call   beeper ; BEEP !
000088CD E950FEFFFF            <1>         jmp     loc_enter_hour_2
                               <1> loc_set_time_bs_1:
000088D2 E8AEFDFFFF            <1>       call   write_backspace
000088D7 E913FEFFFF            <1>         jmp     loc_enter_hour_1
                               <1> loc_set_time_stc_2:
```

```
1012 000088DC E8B6FDFFFF          <1>       call   check_for_backspace
1013 000088E1 740A                <1>       je     short loc_set_time_bs_2
1014                              <1>       ;xor   bh, bh ; video page 0
1015 000088E3 E8F99AFFFF          <1>       call   beeper ; BEEP !
1016 000088E8 E971FEFFFF          <1>         jmp     loc_enter_time_separator_1
1017                              <1> loc_set_time_bs_2:
1018 000088ED E893FDFFFF          <1>       call   write_backspace
1019 000088F2 E92BFEFFFF          <1>         jmp     loc_enter_hour_2
1020                              <1> loc_set_time_stc_3:
1021 000088F7 E89BFDFFFF          <1>       call   check_for_backspace
1022 000088FC 740A                <1>       je     short loc_set_time_bs_3
1023                              <1>       ;xor   bh, bh ; video page 0
1024 000088FE E8DE9AFFFF          <1>       call   beeper ; BEEP !6
1025 00008903 E974FEFFFF          <1>         jmp     loc_enter_minute_1
1026                              <1> loc_set_time_bs_3:
1027 00008908 E878FDFFFF          <1>       call   write_backspace
1028 0000890D E94CFEFFFF          <1>         jmp     loc_enter_time_separator_1
1029                              <1> loc_set_time_stc_4:
1030 00008912 E880FDFFFF          <1>       call   check_for_backspace
1031 00008917 740A                <1>       je     short loc_set_time_bs_4
1032                              <1>       ;xor   bh, bh ; video page 0
1033 00008919 E8C39AFFFF          <1>       call   beeper ; BEEP !
1034 0000891E E984FEFFFF          <1>         jmp     loc_enter_minute_2
1035                              <1> loc_set_time_bs_4:
1036 00008923 E85DFDFFFF          <1>       call   write_backspace
1037 00008928 E94FFEFFFF          <1>         jmp     loc_enter_minute_1
1038                              <1> loc_set_time_stc_5:
1039 0000892D E865FDFFFF          <1>       call   check_for_backspace
1040 00008932 740A                <1>       je     short loc_set_time_bs_5
1041                              <1>       ;xor   bh, bh ; video page 0
1042 00008934 E8A89AFFFF          <1>       call   beeper ; BEEP !
1043 00008939 E994FEFFFF          <1>         jmp     loc_enter_time_separator_2
1044                              <1> loc_set_time_bs_5:
1045 0000893E E842FDFFFF          <1>       call   write_backspace
1046 00008943 E95FFEFFFF          <1>         jmp     loc_enter_minute_2
1047                              <1> loc_set_time_stc_6:
1048 00008948 E84AFDFFFF          <1>       call   check_for_backspace
1049 0000894D 7413                <1>       je     short loc_set_time_bs_6
1050                              <1>       ;xor   bh, bh ; video page 0
1051 0000894F E88D9AFFFF          <1>       call   beeper ; BEEP !
1052 00008954 66C705[96390100]30- <1>       mov    word [Second], 3030h
1052 0000895C 30                  <1>
1053 0000895D E99FFEFFFF          <1>         jmp     loc_enter_second_1
1054                              <1> loc_set_time_bs_6:
1055 00008962 E81EFDFFFF          <1>       call   write_backspace
1056 00008967 E966FEFFFF          <1>         jmp     loc_enter_time_separator_2
1057                              <1> loc_set_time_stc_7:
1058 0000896C E826FDFFFF          <1>       call   check_for_backspace
1059 00008971 740A                <1>       je     short loc_set_time_bs_7
1060                              <1>       ;xor   bh, bh ; video page 0
1061 00008973 E8699AFFFF          <1>       call   beeper ; BEEP !
1062 00008978 E9B3FEFFFF          <1>         jmp     loc_enter_second_2
1063                              <1> loc_set_time_bs_7:
1064 0000897D E803FDFFFF          <1>       call   write_backspace
1065 00008982 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 00008987 88C4                <1>       mov    ah, al
1084 00008989 28C0                <1>       sub    al, al
1085 0000898B 0FB7F0              <1>       movzx  esi, ax
1086 0000898E 81C600010900        <1>       add    esi, Logical_DOSDisks
1087 00008994 8A06                <1>       mov    al, [esi]
1088 00008996 3C41                <1>       cmp    al, 'A'
1089 00008998 7304                <1>       jnb    short loc_pvi_set_vol_name
1090 0000899A 8A6604              <1>       mov    ah, [esi+LD_FSType]
1091 0000899D C3                  <1>       retn
1092                              <1>
1093                              <1> loc_pvi_set_vol_name:
1094 0000899E A2[CA390100]        <1>       mov    [Vol_Drv_Name], al
1095 000089A3 56                  <1>       push   esi
1096 000089A4 E858010000          <1>       call   move_volume_name_and_serial_no ;;;
1097 000089A9 7302                <1>       jnc    short loc_pvi_mvn_ok
1098 000089AB 5E                  <1>       pop    esi
1099 000089AC C3                  <1>       retn
1100                              <1>
1101                              <1> loc_pvi_mvn_ok:
1102 000089AD 8B3424              <1>       mov    esi, [esp]
1103 000089B0 807E04A1            <1>       cmp    byte [esi+LD_FSType], 0A1h
1104 000089B4 7509                <1>       jne    short loc_pvi_fat_vol_size
1105 000089B6 8B4670              <1>       mov    eax, [esi+LD_FS_VolumeSize]
1106 000089B9 0FB75E11            <1>       movzx  ebx, word [esi+LD_FS_BytesPerSec]
1107 000089BD EB07                <1>       jmp    short loc_vol_size_mul32
1108                              <1> loc_pvi_fat_vol_size:
1109 000089BF 8B4670              <1>       mov    eax, [esi+LD_TotalSectors]
1110 000089C2 0FB75E11            <1>       movzx  ebx, word [esi+LD_BPB+BPB_BytsPerSec]
1111                              <1> loc_vol_size_mul32:
1112 000089C6 F7E3                <1>       mul    ebx
1113 000089C8 09D2                <1>       or     edx, edx
1114 000089CA 7507                <1>       jnz    short loc_vol_size_in_kbytes
1115                              <1> loc_vol_size_in_bytes:
```

```
1116 000089CC B9[A8390100]        <1>       mov    ecx, VolSize_Bytes
1117 000089D1 EB0D                <1>       jmp    short loc_write_vol_size_str
1118                              <1> loc_vol_size_in_kbytes:
1119 000089D3 66BB0004            <1>       mov    bx, 1024
1120 000089D7 F7F3                <1>       div    ebx
1121 000089D9 B9[9B390100]        <1>       mov    ecx, VolSize_KiloBytes
1122 000089DE 31D2                <1>       xor    edx, edx ; 0
1123                              <1> loc_write_vol_size_str:
1124 000089E0 890D[8B890100]      <1>       mov    [VolSize_Unit1], ecx
1125                              <1>       ;
1126 000089E6 BF[A1890100]        <1>       mov    edi, Vol_Tot_Sec_Str_End
1127                              <1>         ;mov byte [edi], 0
1128 000089EB B90A000000          <1>       mov    ecx, 10
1129                              <1> loc_write_vol_size_chr:
1130 000089F0 F7F1                <1>       div    ecx
1131 000089F2 80C230              <1>       add    dl, '0'
1132 000089F5 4F                  <1>       dec    edi
1133 000089F6 8817                <1>       mov    [edi], dl
1134 000089F8 85C0                <1>       test   eax, eax
1135 000089FA 7404                <1>       jz     short loc_write_vol_size_str_ok
1136 000089FC 28D2                <1>       sub    dl, dl ; 0
1137 000089FE EBF0                <1>       jmp    short loc_write_vol_size_chr
1138                              <1>
1139                              <1> loc_write_vol_size_str_ok:
1140 00008A00 893D[93890100]      <1>       mov    [Vol_Tot_Sec_Str_Start], edi
1141                              <1>       ;
1142 00008A06 BF[B3390100]        <1>       mov    edi, Vol_FS_Name
1143 00008A0B 8A4E03              <1>       mov    cl, [esi+LD_FATType]
1144 00008A0E 20C9                <1>       and    cl, cl ; 0 ?
1145 00008A10 7515                <1>       jnz    short loc_write_vol_FAT_str_1
1146 00008A12 66C7075452          <1>       mov    word [edi], 'TR'
1147 00008A17 C7470420465331      <1>       mov    dword [edi+4], ' FS1'
1148                              <1>       ;movzx ebx, word [esi+LD_FS_BytesPerSec]
1149 00008A1E 668B5E11            <1>       mov    bx, [esi+LD_FS_BytesPerSec]
1150 00008A22 8B4674              <1>       mov    eax, [esi+LD_FS_FreeSectors]
1151 00008A25 EB36                <1>       jmp    short loc_vol_freespace_mul32
1152                              <1>
1153                              <1> loc_write_vol_FAT_str_1:
1154 00008A27 66B83332            <1>       mov    ax, '32' ; FAT32
1155 00008A2B 80F902              <1>       cmp    cl, 2 ; [esi+LD_FATType]
1156 00008A2E 7708                <1>       ja     short loc_write_vol_FAT_str_2
1157 00008A30 66B83132            <1>       mov    ax, '12' ; FAT12
1158 00008A34 7202                <1>       jb     short loc_write_vol_FAT_str_2
1159 00008A36 B436                <1>       mov    ah, '6'  ; FAT16
1160                              <1> loc_write_vol_FAT_str_2:
1161 00008A38 C70746415420        <1>       mov    dword [edi], 'FAT '
1162 00008A3E 66894704            <1>       mov    word [edi+4], ax
1163                              <1>       ;
1164                              <1>       ;movzx ebx, word [esi+LD_BPB+BPB_BytsPerSec]
1165 00008A42 668B5E11            <1>       mov    bx, [esi+LD_BPB+BPB_BytsPerSec]
1166 00008A46 8B4674              <1>       mov    eax, [esi+LD_FreeSectors]
1167                              <1>
1168                              <1> loc_vol_freespace_recalc0:
1169                              <1>       ; 01/03/2016
1170 00008A49 83F8FF              <1>       cmp    eax, 0FFFFFFFFh
1171 00008A4C 720F                <1>       jb     short loc_vol_freespace_mul32
1172                              <1>       ;inc   eax ; 0
1173 00008A4E 20C9                <1>       and    cl, cl ; byte [esi+LD_FATType]
1174 00008A50 740B                <1>       jz     short loc_vol_freespace_mul32
1175 00008A52 53                  <1>       push   ebx
1176 00008A53 66BB00FF            <1>       mov    bx, 0FF00h ; recalculate free sectors
1177 00008A57 E876490000          <1>       call   calculate_fat_freespace
1178 00008A5C 5B                  <1>       pop    ebx
1179                              <1>
1180                              <1> loc_vol_freespace_mul32:
1181 00008A5D F7E3                <1>       mul    ebx
1182 00008A5F 09D2                <1>       or     edx, edx
1183 00008A61 7507                <1>       jnz    short loc_vol_fspace_in_kbytes
1184                              <1> loc_vol_fspace_in_bytes:
1185 00008A63 B9[A8390100]        <1>       mov    ecx, VolSize_Bytes
1186 00008A68 EB0D                <1>       jmp    short loc_write_vol_fspace_str
1187                              <1> loc_vol_fspace_in_kbytes:
1188 00008A6A 66BB0004            <1>       mov    bx, 1024
1189 00008A6E F7F3                <1>       div    ebx
1190 00008A70 B9[9B390100]        <1>       mov    ecx, VolSize_KiloBytes
1191 00008A75 31D2                <1>       xor    edx, edx ; 0
1192                              <1> loc_write_vol_fspace_str:
1193 00008A77 890D[8F890100]      <1>       mov    [VolSize_Unit2], ecx
1194                              <1>       ;
1195 00008A7D BF[B1890100]        <1>       mov    edi, Vol_Free_Sectors_Str_End
1196                              <1>         ;mov byte [edi], 0
1197 00008A82 B90A000000          <1>       mov    ecx, 10
1198                              <1> loc_write_vol_fspace_chr:
1199 00008A87 F7F1                <1>       div    ecx
1200 00008A89 80C230              <1>       add    dl, '0'
1201 00008A8C 4F                  <1>       dec    edi
1202 00008A8D 8817                <1>       mov    [edi], dl
1203 00008A8F 85C0                <1>       test   eax, eax
1204 00008A91 7404                <1>       jz     short loc_write_vol_fspace_str_ok
1205 00008A93 28D2                <1>       sub    dl, dl ; 0
1206 00008A95 EBF0                <1>       jmp    short loc_write_vol_fspace_chr
1207                              <1>
1208                              <1> loc_write_vol_fspace_str_ok:
1209 00008A97 893D[A3890100]      <1>       mov    [Vol_Free_Sectors_Str_Start], edi
1210                              <1>       ;
1211 00008A9D BE[B1390100]        <1>       mov    esi, Volume_in_drive
1212 00008AA2 E89EEAFFFF          <1>       call   print_msg
1213 00008AA7 BE[F1390100]        <1>       mov    esi, Vol_Name
1214 00008AAC E894EAFFFF          <1>       call   print_msg
1215 00008AB1 BE[BF440100]        <1>       mov    esi, nextline
1216 00008AB6 E88AEAFFFF          <1>       call   print_msg
1217                              <1>       ;
1218 00008ABB BE[523A0100]        <1>       mov    esi, Vol_Total_Sector_Header
1219 00008AC0 E880EAFFFF          <1>       call   print_msg
1220 00008AC5 8B35[93890100]      <1>       mov    esi, [Vol_Tot_Sec_Str_Start]
```

```
1221 00008ACB E875EAFFFF          <1>       call   print_msg
1222 00008AD0 8B35[8B890100]      <1>       mov    esi, [VolSize_Unit1]
1223 00008AD6 E86AEAFFFF          <1>       call   print_msg
1224                              <1>       ;
1225 00008ADB BE[633A0100]        <1>       mov    esi, Vol_Free_Sectors_Header
1226 00008AE0 E860EAFFFF          <1>       call   print_msg
1227 00008AE5 8B35[A3890100]      <1>       mov    esi, [Vol_Free_Sectors_Str_Start]
1228 00008AEB E855EAFFFF          <1>       call   print_msg
1229 00008AF0 8B35[8F890100]      <1>       mov    esi, [VolSize_Unit2]
1230 00008AF6 E84AEAFFFF          <1>       call   print_msg
1231                              <1>       ;
1232 00008AFB 5E                  <1>       pop    esi
1233                              <1>
1234                              <1>       ;mov   ah, [esi+LD_FSType]
1235                              <1>       ;mov   al, [esi+LD_FATType]
1236 00008AFC 668B4603            <1>       mov    ax, [esi+LD_FATType]
1237                              <1>
1238 00008B00 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 00008B01 BF[F1390100]        <1>       mov    edi, Vol_Name
1256                              <1>
1257                              <1>       ;mov   ah, [esi+LD_FSType]
1258                              <1>       ;mov   al, [esi+LD_FATType]
1259 00008B06 668B4603            <1>       mov    ax, [esi+LD_FATType]
1260 00008B0A 80FCA1              <1>       cmp    ah, 0A1h
1261 00008B0D 7418                <1>       je     short mvn_2
1262 00008B0F 08E4                <1>       or     ah, ah
1263 00008B11 7404                <1>       jz     short mvn_0
1264 00008B13 08C0                <1>       or     al, al
1265 00008B15 7504                <1>       jnz    short mvn_1
1266                              <1> mvn_0:
1267 00008B17 8A06                <1>       mov    al, [esi]
1268 00008B19 F9                  <1>       stc
1269 00008B1A C3                  <1>       retn
1270                              <1> mvn_1:
1271 00008B1B 3C02                <1>       cmp    al, 2
1272 00008B1D 7717                <1>       ja     short mvn_3
1273                              <1>       ;or    al, al
1274                              <1>       ;jz    short mvn_2
1275 00008B1F 8B462D              <1>       mov    eax, [esi+LD_BPB+VolumeID]
1276 00008B22 83C631              <1>       add    esi, LD_BPB+VolumeLabel
1277 00008B25 EB15                <1>       jmp    short mvn_4
1278                              <1> mvn_2:
1279 00008B27 8B4628              <1>       mov    eax, [esi+LD_FS_VolumeSerial]
1280 00008B2A 83C62C              <1>       add    esi, LD_FS_VolumeName
1281 00008B2D B910000000          <1>       mov    ecx, 16
1282 00008B32 F3A5                <1>       rep    movsd
1283 00008B34 EB10                <1>       jmp    short mvn_5
1284                              <1> mvn_3:
1285 00008B36 8B4649              <1>       mov    eax, [esi+LD_BPB+FAT32_VolID]
1286 00008B39 83C64D              <1>       add    esi, LD_BPB+FAT32_VolLab
1287                              <1> mvn_4:
1288 00008B3C B90B000000          <1>       mov    ecx, 11
1289 00008B41 F3A4                <1>       rep    movsb
1290 00008B43 C60700              <1>       mov    byte [edi], 0
1291                              <1> mvn_5:
1292                              <1>       ;mov   [Current_VolSerial], eax
1293 00008B46 E8B8B7FFFF          <1>       call   dwordtohex
1294 00008B4B 8915[463A0100]      <1>       mov    [Vol_Serial1], edx
1295 00008B51 A3[4B3A0100]        <1>       mov    [Vol_Serial2], eax
1296                              <1>       ; ecx = 0
1297 00008B56 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 00008B57 31DB                <1>       xor    ebx, ebx
1310 00008B59 88D7                <1>       mov    bh, dl
1311 00008B5B 3815[22380100]      <1>       cmp    [Last_DOS_DiskNo], dl
1312 00008B61 7304                <1>       jnb    short loc_gvsn_start
1313                              <1> loc_gvsn_stc_retn:
1314 00008B63 31C0                <1>       xor    eax, eax
1315 00008B65 F9                  <1>       stc
1316 00008B66 C3                  <1>         retn
1317                              <1> loc_gvsn_start:
1318 00008B67 56                  <1>       push   esi
1319 00008B68 BE00010900          <1>       mov    esi, Logical_DOSDisks
1320 00008B6D 01DE                <1>       add    esi, ebx
1321 00008B6F 8A5E03              <1>       mov    bl, [esi+LD_FATType]
1322 00008B72 20DB                <1>       and    bl, bl
1323 00008B74 740F                <1>       jz     short loc_gvsn_fs
1324 00008B76 80FB02              <1>       cmp    bl, 2
1325 00008B79 7705                <1>       ja     short loc_gvsn_fat32
```

```
                        <1> loc_gvsn_fat:
00008B7B 83C62D         <1>         add    esi, LD_BPB + VolumeID
00008B7E EB0E           <1>         jmp    short loc_gvsn_return
                        <1> loc_gvsn_fat32:
00008B80 83C649         <1>         add    esi, LD_BPB + FAT32_VolID
00008B83 EB09           <1>         jmp    short loc_gvsn_return
                        <1> loc_gvsn_fs:
00008B85 807E04A1       <1>         cmp    byte [esi+LD_FSType], 0A1h
00008B89 75D8           <1>         jne    short loc_gvsn_stc_retn
00008B8B 83C628         <1>         add    esi, LD_FS_VolumeSerial
                        <1> loc_gvsn_return:
00008B8E 8B06           <1>         mov    eax, [esi]
00008B90 5E             <1>         pop    esi
00008B91 C3             <1>         retn
                        <1>
                        <1> ; CMD_INTR.ASM [ TRDOS Command Interpreter Procedure ]
                        <1> ; 09/11/2011
                        <1> ; 29/01/2005
                        <1>
                        <1> command_interpreter:
                        <1>         ; 16/10/2016
                        <1>         ; 12/10/2016
                        <1>         ; 13/05/2016
                        <1>         ; 07/05/2016
                        <1>         ; 04/03/2016
                        <1>         ; 04/02/2016
                        <1>         ; 03/02/2016
                        <1>         ; 30/01/2016
                        <1>         ; 29/01/2016 (TRDOS 386 = TRDOS 2.0)
                        <1>         ; 15/09/2011
                        <1>         ; 29/01/2005
                        <1>
                        <1>         ; Input: ecx = command word length (CL)
                        <1>         ;      CommandBuffer = Command string offset
                        <1>
00008B92 C605[448A0100]00 <1>       mov    byte [Program_Exit],0
00008B99 80F904         <1>         cmp    cl, 4
00008B9C 0F87B5020000   <1>           ja     c_6
00008BA2 0F8237010000   <1>           jb     c_2
                        <1> c_4:
                        <1>
                        <1> cmp_cmd_exit:
00008BA8 BF[90380100]   <1>         mov    edi, Cmd_Exit
00008BAD E8C2030000     <1>         call   cmp_cmd
00008BB2 7208           <1>         jc     short cmp_cmd_date
                        <1>
00008BB4 C605[448A0100]01 <1>         mov     byte [Program_Exit], 1
00008BBB C3             <1>           retn
                        <1>
                        <1> cmp_cmd_date:
00008BBC B104           <1>         mov    cl, 4
00008BBE BF[AC380100]   <1>         mov    edi, Cmd_Date
00008BC3 E8AC030000     <1>         call   cmp_cmd
00008BC8 720B           <1>           jc   short cmp_cmd_time
                        <1>
00008BCA E8D0F7FFFF     <1>         call   show_date
00008BCF E80FF8FFFF     <1>         call   set_date
00008BD4 C3             <1>         retn
                        <1>
                        <1> cmp_cmd_time:
00008BD5 B104           <1>         mov    cl, 4
00008BD7 BF[B1380100]   <1>         mov    edi, Cmd_Time
00008BDC E893030000     <1>         call   cmp_cmd
00008BE1 720B           <1>         jc     short cmp_cmd_show
                        <1>
00008BE3 E8C6FAFFFF     <1>         call   show_time
00008BE8 E8F8FAFFFF     <1>         call   set_time
00008BED C3             <1>         retn
                        <1>
                        <1> cmp_cmd_show:
00008BEE B104           <1>         mov    cl, 4
00008BF0 BF[C2380100]   <1>         mov    edi, Cmd_Show
00008BF5 E87A030000     <1>         call   cmp_cmd
00008BFA 0F83050A0000   <1>           jnc     show_file
                        <1>
                        <1> cmp_cmd_echo:
00008C00 B104           <1>         mov    cl, 4
00008C02 BF[FE380100]   <1>         mov    edi, Cmd_Echo
00008C07 E868030000     <1>         call   cmp_cmd
00008C0C 7224           <1>         jc     short cmp_cmd_copy
                        <1>
                        <1>         ; 22/11/2017
                        <1>         ; AL = 0
00008C0E 803E20         <1>         cmp    byte [esi], 20h
00008C11 7215           <1>         jb     short cmd_echo_nextline
                        <1>         ; 14/04/2016
00008C13 56             <1>         push   esi
                        <1> cmd_echo_asciiz:
                        <1>         ;inc   esi
                        <1>         ;mov   al, [esi]
                        <1>         ; 22/11/2017
00008C14 AC             <1>         lodsb
00008C15 3C20           <1>         cmp    al, 20h
00008C17 73FB           <1>         jnb    short cmd_echo_asciiz
00008C19 4E             <1>         dec    esi
00008C1A C60600         <1>         mov    byte [esi], 0
00008C1D 5E             <1>         pop    esi
00008C1E 89F7           <1>         mov    edi, esi
00008C20 E820E9FFFF     <1>         call   print_msg
00008C25 C60700         <1>         mov    byte [edi], 0
                        <1> cmd_echo_nextline:
00008C28 BE[2D450100]   <1>         mov    esi, NextLine
                        <1>         ;call  print_msg
                        <1>         ;retn
00008C2D E913E9FFFF     <1>         jmp    print_msg
```

```
                                <1>
                                <1> cmp_cmd_copy:
00008C32 B104                   <1>         mov    cl, 4
00008C34 BF[E5380100]           <1>         mov    edi, Cmd_Copy
00008C39 E836030000             <1>         call   cmp_cmd
00008C3E 0F83CC170000           <1>         jnc    copy_file
                                <1>
                                <1> cmp_cmd_move:
00008C44 B104                   <1>         mov    cl, 4
00008C46 BF[EA380100]           <1>         mov    edi, Cmd_Move
00008C4B E824030000             <1>         call   cmp_cmd
00008C50 0F836E160000           <1>         jnc    move_file
                                <1>
                                <1> cmp_cmd_path:
00008C56 B104                   <1>         mov    cl, 4
00008C58 BF[EF380100]           <1>         mov    edi, Cmd_Path
00008C5D E812030000             <1>         call   cmp_cmd
00008C62 0F83F0190000           <1>         jnc    set_get_path
                                <1>
                                <1> cmp_cmd_beep:
00008C68 B104                   <1>         mov    cl, 4
00008C6A BF[1C390100]           <1>         mov    edi, Cmd_Beep
00008C6F E800030000             <1>         call   cmp_cmd
00008C74 720B                   <1>         jc     short cmp_cmd_find
                                <1>         ; 13/05/2016
00008C76 8A3D[BE810100]         <1>         mov    bh, [ptty] ; [ACTIVE_PAGE]
00008C7C E96097FFFF             <1>         jmp    beeper
                                <1>
                                <1> cmp_cmd_find:
00008C81 B104                   <1>         mov    cl, 4
00008C83 BF[F9380100]           <1>         mov    edi, Cmd_Find
00008C88 E8E7020000             <1>         call   cmp_cmd
00008C8D 0F82C4020000           <1>           jc      cmp_cmd_external
                                <1>
                                <1>         ;call  find_and_list_files
00008C93 E9AF220000             <1>         jmp    find_and_list_files
                                <1>         ;retn
                                <1>
                                <1> c_1:
00008C98 AD                     <1>         lodsd
                                <1> cmp_cmd_help:
00008C99 3C3F                   <1>         cmp    al, '?'
00008C9B 751D                   <1>           jne     short cmp_cmd_remark
                                <1>
00008C9D BE[82380100]           <1>         mov    esi, Command_List
                                <1> cmd_help_next_w:
00008CA2 E89EE8FFFF             <1>         call   print_msg
                                <1>
00008CA7 803E20                 <1>         cmp    byte [esi], 20h ; 0
00008CAA 7232                   <1>         jb     short cmd_help_retn
                                <1>
00008CAC 56                     <1>         push   esi
00008CAD BE[BF440100]           <1>         mov    esi, nextline
00008CB2 E88EE8FFFF             <1>         call   print_msg
00008CB7 5E                     <1>         pop    esi
00008CB8 EBE8                   <1>         jmp    short cmd_help_next_w
                                <1>
                                <1> cmp_cmd_remark:
00008CBA 3C2A                   <1>         cmp    al, '*'
00008CBC 0F8595020000           <1>           jne     cmp_cmd_external
00008CC2 46                     <1>         inc    esi
00008CC3 BF[B8820100]           <1>         mov    edi, Remark
00008CC8 8A06                   <1>         mov    al, [esi]
00008CCA 3C20                   <1>         cmp    al, 20h
00008CCC 7707                   <1>         ja     short cmd_remark_write
00008CCE 89FE                   <1>         mov    esi, edi ; Remark
00008CD0 E970E8FFFF             <1>         jmp    print_msg
                                <1>
                                <1> cmd_remark_write:
00008CD5 AA                     <1>         stosb
00008CD6 AC                     <1>         lodsb
00008CD7 3C20                   <1>         cmp    al, 20h
00008CD9 73FA                   <1>         jnb    short cmd_remark_write
00008CDB C60700                 <1>         mov    byte [edi], 0
                                <1>
                                <1> cmd_help_retn:
                                <1> cmd_remark_retn:
                                <1> cd_retn:
00008CDE C3                     <1>         retn
                                <1>
                                <1> c_2:
00008CDF 80F902                 <1>         cmp    cl, 2
00008CE2 0F87AF000000           <1>           ja      c_3
00008CE8 BE[06830100]           <1>         mov    esi, CommandBuffer
00008CED 72A9                   <1>         jb     short c_1
                                <1>
                                <1> cmp_cmd_cd:
00008CEF 66AD                   <1>         lodsw
00008CF1 663D4344               <1>         cmp    ax, 'CD'
00008CF5 7551                   <1>         jne    short cmp_cmd_drive
00008CF7 46                     <1>           inc  esi
                                <1> cd_0:
00008CF8 668B06                 <1>         mov    ax, [esi]
00008CFB 3C20                   <1>         cmp    al, 20h
00008CFD 76DF                   <1>         jna    short cd_retn
                                <1>         ; 10/02/2016
00008CFF 80FC3A                 <1>         cmp    ah, ':'
00008D02 7504                   <1>         jne    short cd_1
00008D04 46                     <1>         inc    esi
00008D05 46                     <1>         inc    esi
00008D06 EB49                   <1>         jmp    short cd_2
                                <1>
                                <1> cd_1: ; change current directory
                                <1>         ; 29/11/2009
                                <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 00008D08 B4CD                     <1>       mov   ah, 0CDh ; mov byte [CD_COMMAND], 0CDh
1541                                    <1>
1542 00008D0A E81D230000               <1>       call  change_current_directory
1543 00008D0F 0F8337220000             <1>         jnc     change_prompt_dir_string
1544                                    <1>
1545                                    <1> cd_error_messages:
1546 00008D15 3C03                     <1>       cmp   al, 3
1547 00008D17 740C                     <1>       je    short cd_path_not_found
1548                                    <1>       ; 16/10/2016 (15h -> 15)
1549 00008D19 3C0F                     <1>       cmp   al, 15 ; drive not ready error
1550 00008D1B 7459                     <1>       je    short cd_drive_not_ready
1551 00008D1D 3C11                     <1>       cmp   al, 17 ; read error
1552 00008D1F 7455                     <1>       je    short cd_drive_not_ready
1553 00008D21 3C13                     <1>       cmp   al, 19 ; ; Bad directory/path name
1554 00008D23 7466                     <1>       je    short cd_command_failed
1555                                    <1>
1556                                    <1> cd_path_not_found:
1557 00008D25 50                       <1>       push  eax ; 29/12/2017
1558                                    <1>       ;push ax
1559 00008D26 BE[253B0100]             <1>       mov   esi, Msg_Dir_Not_Found
1560 00008D2B E815E8FFFF               <1>       call  print_msg
1561                                    <1>       ;pop  ax
1562 00008D30 58                       <1>       pop   eax ; 29/12/2017
1563 00008D31 3A25[54820100]           <1>       cmp   ah, [Current_Dir_Level]
1564 00008D37 0F830F220000             <1>         jnb     change_prompt_dir_string
1565 00008D3D 8825[54820100]           <1>       mov   [Current_Dir_Level], ah
1566 00008D43 E904220000               <1>         jmp     change_prompt_dir_string
1567                                    <1>
1568                                    <1> cmp_cmd_drive: ; change current drive
1569                                    <1>       ; C:, D:, E: etc.
1570 00008D48 80FC3A                   <1>       cmp   ah, ':'
1571 00008D4B 0F8506020000             <1>         jne     cmp_cmd_external
1572                                    <1>
1573                                    <1> cd_2: ; 'CD C:', 'CD D:' ...
1574 00008D51 803E20                   <1>       cmp   byte [esi], 20h
1575 00008D54 0F8707020000             <1>         ja      loc_cmd_failed
1576                                    <1>
1577 00008D5A 24DF                     <1>       and   al, 0DFh
1578 00008D5C 2C41                     <1>       sub   al, 'A'
1579 00008D5E 0F82FD010000             <1>         jc      loc_cmd_failed
1580                                    <1>
1581 00008D64 3A05[22380100]           <1>         cmp     al, [Last_DOS_DiskNo]
1582 00008D6A 770A                     <1>         ja  short cd_drive_not_ready
1583                                    <1>
1584 00008D6C 88C2                     <1>       mov   dl, al
1585 00008D6E E85BF3FFFF               <1>       call  change_current_drive
1586 00008D73 7201                     <1>       jc    short cd_drive_not_ready
1587 00008D75 C3                       <1>       retn
1588                                    <1>
1589                                    <1> cd_drive_not_ready:
1590 00008D76 BE[E23A0100]             <1>       mov   esi, Msg_Not_Ready_Read_Err
1591 00008D7B E8C5E7FFFF               <1>       call  print_msg
1592                                    <1>
1593                                    <1> cd_fail_drive_restart:
1594 00008D80 8A15[56820100]           <1>       mov   dl, [Current_Drv]
1595                                    <1>       ;call  change_current_drive
1596 00008D86 E943F3FFFF               <1>         jmp     change_current_drive
1597                                    <1>       ;retn
1598                                    <1>
1599                                    <1> cd_command_failed:
1600 00008D8B BE[C33A0100]             <1>       mov   esi, Msg_Bad_Command
1601 00008D90 E8B0E7FFFF               <1>       call  print_msg
1602 00008D95 EBE9                     <1>       jmp   short cd_fail_drive_restart
1603                                    <1>
1604                                    <1> c_3:
1605                                    <1> cmp_cmd_dir:
1606 00008D97 BF[82380100]             <1>       mov   edi, Cmd_Dir
1607 00008D9C E8D3010000               <1>       call  cmp_cmd
1608 00008DA1 0F8380020000             <1>       jnc   print_directory_list
1609                                    <1>
1610                                    <1> cmp_cmd_cls:
1611 00008DA7 B103                     <1>       mov   cl, 3
1612 00008DA9 BF[BE380100]             <1>       mov   edi, Cmd_Cls
1613 00008DAE E8C1010000               <1>       call  cmp_cmd
1614 00008DB3 0F83A2E7FFFF             <1>         jnc clear_screen
1615                                    <1>
1616                                    <1> cmp_cmd_ver:
1617 00008DB9 B103                     <1>       mov   cl, 3
1618 00008DBB BF[8C380100]             <1>       mov   edi, Cmd_Ver
1619 00008DC0 E8AF010000               <1>       call  cmp_cmd
1620 00008DC5 720A                     <1>       jc    short cmp_cmd_mem
1621                                    <1>
1622 00008DC7 BE[2A380100]             <1>       mov   esi, mainprog_Version
1623                                    <1>       ;call  print_msg
1624 00008DCC E974E7FFFF               <1>       jmp   print_msg
1625                                    <1>       ;retn
1626                                    <1>
1627                                    <1> cmp_cmd_mem:
1628 00008DD1 B103                     <1>       mov   cl, 3
1629 00008DD3 BF[F4380100]             <1>       mov   edi, Cmd_Mem
1630 00008DD8 E897010000               <1>       call  cmp_cmd
1631 00008DDD 0F8352B4FFFF             <1>       jnc   memory_info
1632                                    <1>
1633                                    <1> cmp_cmd_del:
1634 00008DE3 B103                     <1>       mov   cl, 3
1635 00008DE5 BF[C7380100]             <1>       mov   edi, Cmd_Del
1636 00008DEA E885010000               <1>       call  cmp_cmd
1637 00008DEF 0F83280F0000             <1>         jnc     delete_file
1638                                    <1>
1639                                    <1> cmp_cmd_set:
1640 00008DF5 B103                     <1>       mov   cl, 3
```

```
1641 00008DF7 BF[BA380100]        <1>       mov    edi, Cmd_Set
1642 00008DFC E873010000          <1>       call   cmp_cmd
1643 00008E01 0F83C9170000        <1>         jnc     set_get_env
1644                              <1>
1645                              <1> cmp_cmd_run:
1646 00008E07 B103                <1>       mov    cl, 3
1647 00008E09 BF[B6380100]        <1>       mov    edi, Cmd_Run
1648 00008E0E E861010000          <1>       call   cmp_cmd
1649                              <1>       ; 07/05/2016
1650 00008E13 0F823E010000        <1>         jc      cmp_cmd_external
1651 00008E19 E90F1E0000          <1>       jmp    load_and_execute_file
1652                              <1> c_5:
1653                              <1> cmp_cmd_mkdir:
1654 00008E1E BF[DF380100]        <1>       mov    edi, Cmd_Mkdir
1655 00008E23 E84C010000          <1>       call   cmp_cmd
1656 00008E28 0F83990A0000        <1>         jnc     make_directory
1657                              <1>
1658                              <1> cmp_cmd_rmdir:
1659 00008E2E B105                <1>       mov    cl, 5
1660 00008E30 BF[D9380100]        <1>       mov    edi, Cmd_Rmdir
1661 00008E35 E83A010000          <1>       call   cmp_cmd
1662 00008E3A 0F83AA0B0000        <1>         jnc     delete_directory
1663                              <1>
1664                              <1> cmp_cmd_chdir:
1665 00008E40 B105                <1>       mov    cl, 5
1666 00008E42 BF[16390100]        <1>       mov    edi, Cmd_Chdir
1667 00008E47 E828010000          <1>       call   cmp_cmd
1668 00008E4C 0F8205010000        <1>         jc      cmp_cmd_external
1669                              <1>
1670 00008E52 E9A1FEFFFF          <1>       jmp    cd_0
1671                              <1>
1672                              <1> c_6:
1673 00008E57 80F906              <1>       cmp    cl, 6
1674 00008E5A 0F87E0000000        <1>         ja      c_8
1675 00008E60 72BC                <1>       jb     short c_5
1676                              <1> cmp_cmd_prompt:
1677 00008E62 BF[95380100]        <1>       mov    edi, Cmd_Prompt
1678 00008E67 E808010000          <1>       call   cmp_cmd
1679 00008E6C 722F                <1>         jc   short cmp_cmd_volume
1680                              <1> get_prompt_name_fchar:
1681 00008E6E AC                  <1>       lodsb
1682 00008E6F 3C20                <1>       cmp    al, 20h
1683 00008E71 74FB                <1>       je     short get_prompt_name_fchar
1684 00008E73 7713                <1>       ja     short loc_change_prompt_label
1685                              <1> default_command_prompt: ; 31/12/2017 ('sysprompt')
1686 00008E75 BE[76380100]        <1>       mov    esi, TRDOSPromptLabel
1687 00008E7A C7065452444F        <1>       mov    dword [esi], "TRDO"
1688 00008E80 66C746045300        <1>              mov    word [esi+4], "S"
1689                              <1> loc_cmd_prompt_return:
1690 00008E86 C3                  <1>       retn
1691                              <1>
1692                              <1> set_command_prompt: ; 31/12/2017 ('sysprompt')
1693 00008E87 AC                  <1>       lodsb
1694                              <1> loc_change_prompt_label:
1695 00008E88 66B90B00            <1>       mov    cx, 11
1696 00008E8C BF[76380100]        <1>       mov    edi, TRDOSPromptLabel
1697                              <1> put_char_new_prompt_label:
1698 00008E91 AA                  <1>       stosb
1699 00008E92 AC                  <1>       lodsb
1700 00008E93 3C20                <1>       cmp    al, 20h
1701 00008E95 7202                <1>       jb     short pass_put_new_prompt_label
1702 00008E97 E2F8                <1>       loop   put_char_new_prompt_label
1703                              <1> pass_put_new_prompt_label:
1704 00008E99 C60700              <1>       mov    byte [edi], 0
1705 00008E9C C3                  <1>       retn
1706                              <1>
1707                              <1> cmp_cmd_volume:
1708 00008E9D B106                <1>       mov    cl, 6
1709 00008E9F BF[9C380100]        <1>       mov    edi, Cmd_Volume
1710 00008EA4 E8CB000000          <1>       call   cmp_cmd
1711 00008EA9 7255                <1>         jc   short cmp_cmd_attrib
1712                              <1>
1713                              <1> cmd_vol1:
1714 00008EAB AC                  <1>       lodsb
1715 00008EAC 3C20                <1>       cmp    al, 20h
1716 00008EAE 7707                <1>       ja     short cmd_vol2
1717 00008EB0 A0[56820100]        <1>       mov    al, [Current_Drv]
1718 00008EB5 EB3D                <1>       jmp    short cmd_vol4
1719                              <1> cmd_vol2:
1720 00008EB7 3C41                <1>       cmp    al, 'A'
1721 00008EB9 0F82A2000000        <1>         jb      loc_cmd_failed
1722 00008EBF 3C7A                <1>       cmp    al, 'z'
1723 00008EC1 0F879A000000        <1>         ja      loc_cmd_failed
1724 00008EC7 3C5A                <1>       cmp    al, 'Z'
1725 00008EC9 760A                <1>       jna    short cmd_vol3
1726 00008ECB 3C61                <1>       cmp    al, 'a'
1727 00008ECD 0F828E000000        <1>         jb      loc_cmd_failed
1728 00008ED3 24DF                <1>       and    al, 0DFh
1729                              <1> cmd_vol3:
1730 00008ED5 8A26                <1>       mov    ah, [esi]
1731 00008ED7 80FC3A              <1>       cmp    ah, ':'
1732 00008EDA 0F8581000000        <1>         jne     loc_cmd_failed
1733 00008EE0 2C41                <1>       sub    al, 'A'
1734 00008EE2 3A05[22380100]      <1>         cmp     al, [Last_DOS_DiskNo]
1735 00008EE8 760A                <1>       jna    short cmd_vol4
1736                              <1>
1737 00008EEA BE[E23A0100]        <1>       mov    esi, Msg_Not_Ready_Read_Err
1738 00008EEF E951E6FFFF          <1>       jmp    print_msg
1739                              <1>
1740                              <1> cmd_vol4:
1741 00008EF4 E88EFAFFFF          <1>       call   print_volume_info
1742 00008EF9 0F8277FEFFFF        <1>         jc      cd_drive_not_ready
1743 00008EFF C3                  <1>       retn
1744                              <1>
1745                              <1> cmp_cmd_attrib:
```

```
1746 00008F00 B106              <1>       mov    cl, 6
1747 00008F02 BF[CB380100]      <1>       mov    edi, Cmd_Attrib
1748 00008F07 E868000000        <1>       call   cmp_cmd
1749 00008F0C 0F831D0F0000      <1>         jnc     set_file_attributes
1750                            <1>
1751                            <1> cmp_cmd_rename:
1752 00008F12 B106              <1>       mov    cl, 6
1753 00008F14 BF[D2380100]      <1>       mov    edi, Cmd_Rename
1754 00008F19 E856000000        <1>       call   cmp_cmd
1755 00008F1E 0F8353110000      <1>         jnc     rename_file
1756                            <1>
1757                            <1> cmp_cmd_device:
1758 00008F24 B106              <1>       mov    cl, 6
1759 00008F26 BF[07390100]      <1>       mov    edi, Cmd_Device
1760 00008F2B E844000000        <1>       call   cmp_cmd
1761 00008F30 7225              <1>         jc   short cmp_cmd_external
1762                            <1>
1763 00008F32 C3                <1>       retn
1764                            <1>
1765                            <1> c_7:
1766                            <1> cmp_cmd_devlist:
1767 00008F33 BF[0E390100]      <1>       mov    edi, Cmd_DevList
1768 00008F38 E837000000        <1>       call   cmp_cmd
1769 00008F3D 7218              <1>         jc   short cmp_cmd_external
1770                            <1>
1771                            <1> loc_cmd_return:
1772 00008F3F C3                <1>       retn
1773                            <1>
1774                            <1> c_8:
1775 00008F40 80F908            <1>         cmp  cl, 8
1776 00008F43 7712              <1>       ja     short cmp_cmd_external
1777 00008F45 72EC              <1>       jb     short c_7
1778                            <1>
1779                            <1> cmp_cmd_longname:
1780 00008F47 BF[A3380100]      <1>       mov    edi, Cmd_LongName
1781 00008F4C E823000000        <1>       call   cmp_cmd
1782 00008F51 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 00008F57 BE[06830100]      <1>       mov    esi, CommandBuffer
1788 00008F5C E9CC1C0000        <1>       jmp    loc_run_check_filename
1789                            <1>
1790                            <1> loc_cmd_failed:
1791 00008F61 803D[06830100]20  <1>       cmp    byte [CommandBuffer], 20h
1792 00008F68 76D5              <1>       jna    short loc_cmd_return
1793 00008F6A BE[C33A0100]      <1>       mov    esi, Msg_Bad_Command
1794                            <1> ;     call   print_msg
1795                            <1> ;loc_cmd_return:
1796                            <1> ;     retn
1797 00008F6F E9D1E5FFFF        <1>       jmp    print_msg
1798                            <1>
1799                            <1> cmp_cmd:
1800                            <1>        ; 29/01/2016 (TRDOS 386 = TRDOS v2.0)
1801 00008F74 BE[06830100]      <1>         mov esi, CommandBuffer
1802                            <1>         ; edi = internal command word (ASCIIZ)
1803                            <1>        ; ecx = command length (<=8)
1804                            <1> cmp_cmd_1:
1805 00008F79 AC                <1>       lodsb
1806 00008F7A AE                <1>       scasb
1807 00008F7B 750D              <1>       jne    short cmp_cmd_3
1808 00008F7D E2FA              <1>       loop   cmp_cmd_1
1809 00008F7F AC                <1>       lodsb
1810 00008F80 3C20              <1>       cmp    al, 20h
1811 00008F82 7703              <1>       ja     short cmp_cmd_2
1812 00008F84 30C0              <1>       xor    al, al
1813                            <1>       ; ZF = 1 -> internal command word matches
1814 00008F86 C3                <1>       retn
1815                            <1> cmp_cmd_2:
1816                            <1>       ; ZF = 0 (CF = 0) -> external command word
1817 00008F87 58                <1>       pop    eax ; no return to the caller from here
1818 00008F88 EBCD              <1>       jmp    cmp_cmd_external
1819                            <1> cmp_cmd_3:
1820 00008F8A F9                <1>       stc
1821                            <1>       ; CF = 1 -> internal command word does not match
1822 00008F8B 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 00008F8C 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 00008F91 3C01              <1>       cmp    al, 1 ; Bad command or file name
1836 00008F93 74CC              <1>       je     loc_cmd_failed
1837                            <1> loc_run_dir_not_found:
1838 00008F95 3C03              <1>       cmp    al, 3
1839 00008F97 750A              <1>       jne    short loc_run_file_notfound_msg
1840                            <1>       ; Path not found (MS-DOS Error Code = 3)
1841 00008F99 BE[253B0100]      <1>       mov    esi, Msg_Dir_Not_Found
1842 00008F9E E9A2E5FFFF        <1>       jmp    print_msg
1843                            <1>
1844                            <1> loc_run_file_notfound_msg:
1845 00008FA3 3C02              <1>       cmp    al, 2 ; File not found
1846 00008FA5 750A              <1>       jne    short loc_run_file_drv_read_err
1847                            <1>
1848                            <1> loc_print_file_notfound_msg:
1849 00008FA7 BE[3C3B0100]      <1>         mov     esi, Msg_File_Not_Found
1850                            <1>       ;call  proc_printmsg
```

```
1851                            <1>       ;retn
1852 00008FAC E994E5FFFF          <1>       jmp    print_msg
1853                            <1>
1854                            <1> loc_run_file_drv_read_err:
1855                            <1>       ; Err: 17 (Read fault)
1856 00008FB1 3C11                <1>       cmp    al, 17 ; Drive not ready or read error
1857 00008FB3 7404                <1>       je     short loc_run_file_print_drv_read_err
1858                            <1>       ;
1859 00008FB5 3C0F                <1>       cmp    al, 15 ; Drive not ready (or read error)
1860 00008FB7 750A                <1>       jne    short loc_run_file_toobig
1861                            <1>
1862                            <1> loc_run_file_print_drv_read_err:
1863 00008FB9 BE[E23A0100]        <1>       mov    esi, Msg_Not_Ready_Read_Err
1864 00008FBE E982E5FFFF          <1>       jmp    print_msg
1865                            <1>
1866                            <1> loc_run_file_toobig:
1867 00008FC3 3C08                <1>       cmp    al, 8 ; Not enough free memory to load&run file
1868 00008FC5 750A                <1>       jne    short loc_run_file_perm_denied
1869 00008FC7 BE[873B0100]        <1>       mov    esi, Msg_Insufficient_Memory
1870 00008FCC E974E5FFFF          <1>       jmp    print_msg
1871                            <1>
1872                            <1> loc_run_file_perm_denied:
1873                            <1>       ; 29/12/2017
1874 00008FD1 3C0B                <1>       cmp    al, ERR_PERM_DENIED ; 11 ; Permission denied
1875 00008FD3 750A                <1>       jne    short loc_run_misc_error
1876 00008FD5 BE[1C3D0100]        <1>       mov    esi, Msg_Permission_Denied
1877 00008FDA 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 00008FDF E8DFB2FFFF          <1>       call   bytetohex
1884 00008FE4 66A3[BB3B0100]      <1>         mov     [error_code_hex], ax
1885                            <1>
1886 00008FEA BE[9E3B0100]        <1>       mov    esi, Msg_Error_Code
1887                            <1>       ;call  print_msg
1888                            <1>       ;retn
1889                            <1>
1890 00008FEF 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 00008FF4 50                  <1>       push   eax
1895 00008FF5 8A3D[B2890100]      <1>       mov    bh, [RUN_CDRV] ; it is set at the beginning
1896                            <1>                              ; of the 'run' command.
1897 00008FFB 3A3D[56820100]      <1>       cmp    bh, [Current_Drv]
1898 00009001 7409                <1>       je     short loc_run_restore_cdir
1899 00009003 88FA                <1>       mov    dl, bh
1900 00009005 E8C4F0FFFF          <1>       call   change_current_drive
1901 0000900A EB19                <1>       jmp    short loc_run_err_pass_restore_cdir
1902                            <1>
1903                            <1> loc_run_restore_cdir:
1904 0000900C 803D[23380100]00    <1>       cmp    byte [Restore_CDIR], 0
1905 00009013 7610                <1>       jna    short loc_run_err_pass_restore_cdir
1906 00009015 30DB                <1>       xor    bl, bl
1907 00009017 0FB7F3              <1>       movzx  esi, bx
1908 0000901A 81C600010900        <1>       add    esi, Logical_DOSDisks
1909 00009020 E860F1FFFF          <1>       call   restore_current_directory
1910                            <1>
1911                            <1> loc_run_err_pass_restore_cdir:
1912 00009025 58                  <1>       pop    eax
1913 00009026 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 00009027 66C705[F48A0100]00- <1>       mov    word [AttributesMask], 0800h ; ..except volume names..
1920 0000902F 08                  <1>
1921 00009030 A0[56820100]        <1>       mov    al, [Current_Drv]
1922 00009035 A2[B2890100]        <1>       mov    [RUN_CDRV], al
1923                            <1> get_dfname_fchar:
1924 0000903A AC                  <1>       lodsb
1925 0000903B 3C20                <1>       cmp    al, 20h
1926 0000903D 74FB                <1>       je     short get_dfname_fchar
1927 0000903F 0F82A4000000        <1>         jb      loc_print_dir_call_all
1928 00009045 3C2D                <1>       cmp    al, '-'
1929 00009047 7542                <1>       jne    short loc_print_dir_call_flt
1930                            <1> get_next_attr_char:
1931 00009049 AC                  <1>       lodsb
1932 0000904A 3C20                <1>       cmp    al, 20h
1933 0000904C 74FB                <1>       je     short get_next_attr_char
1934 0000904E 0F820DFFFFFF        <1>         jb      loc_cmd_failed
1935 00009054 24DF                <1>       and    al, 0DFh
1936 00009056 3C44                <1>       cmp    al, 'D' ; directories only ?
1937 00009058 7512                <1>       jne    short pass_only_directories
1938 0000905A AC                  <1>       lodsb
1939 0000905B 3C20                <1>       cmp    al, 20h
1940 0000905D 0F87FEFEFFFF        <1>         ja      loc_cmd_failed
1941 00009063 800D[F48A0100]10    <1>       or     byte [AttributesMask], 10h ; ..directory..
1942 0000906A EB18                <1>       jmp    short get_dfname_fchar_attr
1943                            <1> pass_only_directories:
1944 0000906C 3C46                <1>       cmp    al, 'F'      ; files only ?
1945 0000906E 0F85B0000000        <1>         jne     check_attr_s
1946 00009074 AC                  <1>       lodsb
1947 00009075 3C20                <1>       cmp    al, 20h
1948 00009077 0F87E4FEFFFF        <1>         ja      loc_cmd_failed
1949 0000907D 800D[F58A0100]10    <1>       or     byte [AttributesMask+1], 10h ; ..except directories..
1950                            <1> get_dfname_fchar_attr:
1951 00009084 AC                  <1>       lodsb
1952 00009085 3C20                <1>       cmp    al, 20h
1953 00009087 74FB                <1>       je     short get_dfname_fchar_attr
1954 00009089 725E                <1>       jb     short loc_print_dir_call_all
```

```
1955                                    <1>
1956                                    <1> loc_print_dir_call_flt:
1957 0000908B 4E                       <1>         dec     esi
1958 0000908C BF[F68A0100]              <1>         mov     edi, FindFile_Drv
1959 00009091 E8AC250000                <1>         call    parse_path_name
1960 00009096 7308                      <1>         jnc     short loc_print_dir_change_drv_1
1961 00009098 3C01                      <1>         cmp     al, 1
1962 0000909A 0F87ECFEFFFF              <1>           ja      loc_run_cmd_failed
1963                                    <1>
1964                                    <1> loc_print_dir_change_drv_1:
1965 000090A0 8A15[F68A0100]            <1>         mov     dl, [FindFile_Drv]
1966                                    <1> loc_print_dir_change_drv_2:
1967 000090A6 3A15[B2890100]            <1>         cmp     dl, [RUN_CDRV]
1968 000090AC 740B                      <1>         je      short loc_print_dir_change_directory
1969 000090AE E81BF0FFFF                <1>         call    change_current_drive
1970 000090B3 0F82D3FEFFFF              <1>           jc      loc_run_cmd_failed
1971                                    <1> loc_print_dir_change_directory:
1972 000090B9 803D[F78A0100]20          <1>         cmp     byte [FindFile_Directory], 20h ; 0 or 20h ?
1973 000090C0 761D                      <1>         jna     short pass_print_dir_change_directory
1974                                    <1>
1975 000090C2 FE05[23380100]            <1>         inc     byte [Restore_CDIR]
1976 000090C8 BE[F78A0100]              <1>         mov     esi, FindFile_Directory
1977 000090CD 30E4                      <1>         xor     ah, ah ; CD_COMMAND sign -> 0
1978 000090CF E8581F0000                <1>         call    change_current_directory
1979 000090D4 0F82B2FEFFFF              <1>           jc      loc_run_cmd_failed
1980                                    <1>
1981                                    <1> loc_print_dir_change_prompt_dir_string:
1982 000090DA E86D1E0000                <1>         call    change_prompt_dir_string
1983                                    <1>
1984                                    <1> pass_print_dir_change_directory:
1985 000090DF BE[388B0100]              <1>         mov     esi, FindFile_Name
1986 000090E4 803E20                    <1>         cmp     byte [esi], 20h ; ; 0 or 20h ?
1987 000090E7 7706                      <1>         ja      short loc_print_dir_call
1988                                    <1>
1989                                    <1> loc_print_dir_call_all:
1990 000090E9 C7062A2E2A00              <1>         mov     dword [esi], '*.*'
1991                                    <1> loc_print_dir_call:
1992 000090EF E87E000000                <1>         call    print_directory
1993                                    <1>
1994 000090F4 8A15[B2890100]            <1>         mov     dl, [RUN_CDRV]  ; it is set at the beginning
1995 000090FA 3A15[56820100]            <1>         cmp     dl, [Current_Drv]
1996 00009100 7406                      <1>         je      short loc_print_dir_call_restore_cdir_retn
1997 00009102 E8C7EFFFFF                <1>         call    change_current_drive
1998 00009107 C3                        <1>         retn
1999                                    <1>
2000                                    <1> loc_print_dir_call_restore_cdir_retn:
2001 00009108 803D[23380100]00          <1>         cmp     byte [Restore_CDIR], 0
2002 0000910F 7610                      <1>         jna     short pass_print_dir_call_restore_cdir_retn
2003                                    <1>
2004 00009111 BE00010900                <1>         mov     esi, Logical_DOSDisks
2005 00009116 31C0                      <1>         xor     eax, eax
2006 00009118 88D4                      <1>         mov     ah, dl
2007 0000911A 01C6                      <1>         add     esi, eax
2008                                    <1>
2009 0000911C E864F0FFFF                <1>         call    restore_current_directory
2010                                    <1>
2011                                    <1> pass_print_dir_call_restore_cdir_retn:
2012 00009121 C3                        <1>         retn
2013                                    <1>
2014                                    <1> check_attr_s_cap:
2015 00009122 24DF                      <1>         and     al, 0DFh
2016                                    <1> check_attr_s:
2017 00009124 3C53                      <1>         cmp     al, 'S'
2018 00009126 7514                      <1>         jne     short pass_attr_s
2019 00009128 800D[F48A0100]04          <1>         or      byte [AttributesMask], 4 ; system
2020 0000912F AC                        <1>         lodsb
2021 00009130 3C20                      <1>         cmp     al, 20h
2022 00009132 0F844CFFFFFF              <1>           je      get_dfname_fchar_attr
2023 00009138 72AF                      <1>         jb      short loc_print_dir_call_all
2024 0000913A 24DF                      <1>         and     al, 0DFh
2025                                    <1> pass_attr_s:
2026 0000913C 3C48                      <1>         cmp     al, 'H'
2027 0000913E 7514                      <1>         jne     short pass_attr_h
2028 00009140 800D[F48A0100]02          <1>         or      byte [AttributesMask], 2 ; hidden
2029                                    <1> pass_attr_shr:
2030 00009147 AC                        <1>         lodsb
2031 00009148 3C20                      <1>         cmp     al, 20h
2032 0000914A 0F8434FFFFFF              <1>           je      get_dfname_fchar_attr
2033 00009150 7297                      <1>         jb      short loc_print_dir_call_all
2034 00009152 EBCE                      <1>         jmp     short check_attr_s_cap
2035                                    <1>
2036                                    <1> pass_attr_h:
2037 00009154 3C52                      <1>         cmp     al, 'R'
2038 00009156 7509                      <1>         jne     short pass_attr_r
2039 00009158 800D[F48A0100]01          <1>         or      byte [AttributesMask], 1 ; read only
2040 0000915F EBE6                      <1>         jmp     short pass_attr_shr
2041                                    <1>
2042                                    <1> pass_attr_r:
2043 00009161 3C41                      <1>         cmp     al, 'A'
2044 00009163 0F85F8FDFFFF              <1>           jne     loc_cmd_failed
2045 00009169 800D[F48A0100]20          <1>         or      byte [AttributesMask], 20h ; archive
2046 00009170 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 00009172 56                       <1>         push    esi
```

```
2060                            <1>
2061 00009173 29C0              <1>       sub    eax, eax
2062                            <1>
2063 00009175 66A3[808B0100]    <1>       mov    word [Dir_Count], ax ; 0
2064 0000917B 66A3[7E8B0100]    <1>       mov    word [File_Count], ax ; 0
2065 00009181 A3[828B0100]      <1>       mov    dword [Total_FSize], eax ; 0
2066                            <1>
2067 00009186 E8D0E3FFFF        <1>       call    clear_screen
2068                            <1>
2069 0000918B 31C9              <1>       xor    ecx, ecx
2070 0000918D 8A2D[56820100]    <1>       mov     ch, [Current_Drv] ; DirBuff_Drv - 'A'
2071 00009193 A0[57820100]      <1>       mov     al, [Current_Dir_Drv]
2072 00009198 A2[E0390100]      <1>       mov     [Dir_Drive_Name], al
2073 0000919D BE00010900        <1>       mov    esi, Logical_DOSDisks
2074 000091A2 01CE              <1>       add    esi, ecx
2075                            <1>
2076 000091A4 E858F9FFFF        <1>       call   move_volume_name_and_serial_no
2077 000091A9 730C              <1>       jnc    short print_dir_strlen_check
2078                            <1>
2079 000091AB 5E                <1>       pop    esi
2080 000091AC 8A3D[BE810100]    <1>       mov    bh, [ptty] ; [ACTIVE_PAGE]
2081                            <1>       ;call  beeper
2082                            <1>       ;retn
2083 000091B2 E92A92FFFF        <1>       jmp    beeper  ; beep ! and return
2084                            <1>
2085                            <1> print_dir_strlen_check:
2086 000091B7 BE[59820100]      <1>       mov    esi, Current_Dir_Root
2087 000091BC BF[7D3A0100]      <1>       mov    edi, Dir_Str_Root
2088                            <1>
2089                            <1>       ;xor   ecx, ecx
2090 000091C1 8A0D[B5820100]    <1>         mov     cl, [Current_Dir_StrLen]
2091 000091C7 FEC1              <1>       inc    cl
2092 000091C9 80F940            <1>       cmp    cl, 64
2093 000091CC 760D              <1>       jna    short pass_print_dir_strlen_shorting
2094 000091CE 46                <1>       inc    esi
2095 000091CF 01CE              <1>       add    esi, ecx
2096 000091D1 83EE40            <1>       sub    esi, 64
2097 000091D4 47                <1>       inc    edi
2098 000091D5 B82E2E2E20        <1>       mov    eax, '... '
2099 000091DA AB                <1>       stosd
2100                            <1>
2101                            <1> pass_print_dir_strlen_shorting:
2102 000091DB F3A4              <1>       rep    movsb
2103                            <1>
2104 000091DD BE[D3390100]      <1>       mov    esi, Dir_Drive_Str
2105 000091E2 E85EE3FFFF        <1>       call   print_msg
2106                            <1>
2107 000091E7 BE[323A0100]      <1>       mov    esi, Vol_Serial_Header
2108 000091EC E854E3FFFF        <1>       call   print_msg
2109                            <1>
2110 000091F1 BE[723A0100]      <1>       mov    esi, Dir_Str_Header
2111 000091F6 E84AE3FFFF        <1>       call   print_msg
2112                            <1>
2113 000091FB BE[BD440100]      <1>       mov    esi, next2line
2114 00009200 E840E3FFFF        <1>       call   print_msg
2115                            <1>
2116                            <1> loc_print_dir_first_file:
2117 00009205 C605[958B0100]10  <1>       mov    byte [PrintDir_RowCounter], 16
2118 0000920C 66A1[F48A0100]    <1>       mov    ax, [AttributesMask]
2119 00009212 5E                <1>       pop    esi
2120                            <1>
2121 00009213 E859020000        <1>       call   find_first_file
2122 00009218 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 0000921E F6C310            <1>       test   bl, 10h  ; Is it a directory?
2127 00009221 741B              <1>       jz     short loc_not_dir
2128                            <1>
2129 00009223 66FF05[808B0100]  <1>       inc    word [Dir_Count]
2130 0000922A 89F2              <1>       mov    edx, esi     ; FindFile_DirEntry address
2131 0000922C BE[C23B0100]      <1>       mov    esi, Type_Dir; '<DIR>     '
2132 00009231 BF[D93B0100]      <1>       mov    edi, Dir_Or_FileSize
2133                            <1>       ; move 10 bytes
2134 00009236 A5                <1>       movsd
2135 00009237 A5                <1>       movsd
2136 00009238 66A5              <1>       movsw
2137 0000923A 89D6              <1>       mov    esi, edx
2138 0000923C EB36              <1>       jmp     short loc_dir_attribute
2139                            <1>
2140                            <1> loc_not_dir:
2141 0000923E 66FF05[7E8B0100]  <1>       inc    word [File_Count]
2142 00009245 0105[828B0100]    <1>       add    [Total_FSize], eax
2143                            <1>
2144 0000924B B90A000000        <1>       mov    ecx, 10  ; 32 bit divisor
2145 00009250 89CF              <1>       mov    edi, ecx
2146 00009252 81C7[D93B0100]    <1>       add    edi, Dir_Or_FileSize
2147                            <1> loc_dir_rdivide:
2148 00009258 29D2              <1>       sub    edx, edx
2149 0000925A F7F1              <1>       div    ecx     ; remainder in dl (< 10)
2150 0000925C 80C230            <1>       add     dl, '0'      ; to make visible (ascii)
2151 0000925F 4F                <1>       dec    edi
2152 00009260 8817              <1>       mov     [edi], dl
2153 00009262 21C0              <1>       and    eax, eax
2154 00009264 75F2              <1>       jnz    short loc_dir_rdivide
2155                            <1>
2156                            <1> loc_dir_fill_space:
2157 00009266 81FF[D93B0100]    <1>       cmp     edi, Dir_Or_FileSize
2158 0000926C 7606              <1>       jna     short loc_dir_attribute
2159 0000926E 4F                <1>       dec     edi
2160 0000926F C60720            <1>       mov     byte [edi], 20h
2161 00009272 EBF2              <1>       jmp     short loc_dir_fill_space
2162                            <1>
2163                            <1> loc_dir_attribute:
2164 00009274 C705[E43B0100]2020- <1>       mov    dword [File_Attribute], 20202020h
```

```
2164 0000927C 2020                <1>
2165                              <1>
2166 0000927E 80FB20              <1>       cmp    bl, 20h  ; Is it an archive file?
2167 00009281 7207                <1>       jb     short loc_dir_pass_arch
2168 00009283 C605[E73B0100]41    <1>       mov    byte [File_Attribute+3], 'A'
2169                              <1>
2170                              <1> loc_dir_pass_arch:
2171 0000928A 80E307              <1>       and    bl, 7
2172 0000928D 7428                <1>       jz     short loc_dir_file_name
2173 0000928F 88DF                <1>       mov    bh, bl
2174 00009291 80E303              <1>       and    bl, 3
2175 00009294 38DF                <1>       cmp    bh, bl
2176 00009296 7607                <1>       jna    short loc_dir_pass_s
2177 00009298 C605[E43B0100]53    <1>       mov    byte [File_Attribute], 'S'
2178                              <1>
2179                              <1> loc_dir_pass_s:
2180 0000929F 80E302              <1>       and     bl,2
2181 000092A2 7407                <1>       jz      short loc_dir_pass_h
2182 000092A4 C605[E53B0100]48    <1>       mov     byte [File_Attribute+1], 'H'
2183                              <1> loc_dir_pass_h:
2184 000092AB 80E701              <1>       and     bh,1
2185 000092AE 7407                <1>       jz      short loc_dir_file_name
2186 000092B0 C605[E63B0100]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 000092B7 8B5E16              <1>       mov    ebx, [esi+16h]
2191 000092BA 89F1                <1>       mov    ecx, esi ; FindFile_DirEntry address
2192 000092BC BF[CC3B0100]        <1>       mov     edi, File_Name
2193                              <1>       ; move 8 bytes
2194 000092C1 A5                  <1>       movsd
2195 000092C2 A5                  <1>       movsd
2196 000092C3 C60720              <1>       mov    byte [edi], 20h
2197 000092C6 47                  <1>       inc    edi
2198                              <1>       ; move 3 bytes
2199 000092C7 66A5                <1>       movsw
2200 000092C9 A4                  <1>       movsb
2201 000092CA 89CE                <1>       mov    esi, ecx
2202                              <1>
2203                              <1> Dir_Time_start:
2204                              <1>       ;mov   ax, dx       ; Time
2205 000092CC 6689D8              <1>       mov    ax, bx
2206 000092CF 66C1E805            <1>       shr    ax, 5        ; shift right 5 times
2207 000092D3 6683E03F            <1>       and    ax, 0000111111b     ; Minute Mask
2208 000092D7 D40A                <1>       aam                 ; Q([AL]/10)->AH
2209                              <1>                           ; R([AL]/10)->AL
2210                              <1>                           ; [AL]+[AH]= Minute as BCD
2211 000092D9 660D3030            <1>       or     ax, '00'     ; Convert to ASCII
2212 000092DD 86E0                <1>       xchg   ah, al
2213 000092DF 66A3[F73B0100]      <1>       mov    [File_Minute], ax
2214                              <1>
2215                              <1>       ;mov   al, dh
2216 000092E5 88F8                <1>       mov    al, bh
2217 000092E7 C0E803              <1>       shr    al, 3        ; shift right 3 times
2218 000092EA D40A                <1>       aam                 ; [AL]+[AH]= Hours as BCD
2219 000092EC 660D3030            <1>       or     ax, '00'
2220 000092F0 86E0                <1>       xchg   ah, al
2221 000092F2 66A3[F43B0100]      <1>       mov     [File_Hour], ax
2222                              <1>
2223 000092F8 C1EB10              <1>       shr    ebx, 16             ; BX = Date
2224                              <1>
2225                              <1> Dir_Date_start:
2226 000092FB 6689D8              <1>       mov    ax, bx       ; Date
2227 000092FE 6683E01F            <1>       and    ax, 00011111b; Day Mask
2228 00009302 D40A                <1>       aam                 ; Q([AL]/10)->AH
2229                              <1>                           ; R([AL]/10)->AL
2230                              <1>                           ; [AL]+[AH]= Day as BCD
2231 00009304 660D3030            <1>       or     ax, '00'     ; Convert to ASCII
2232 00009308 86C4                <1>       xchg   al, ah
2233                              <1>
2234 0000930A 66A3[E93B0100]      <1>       mov    [File_Day], ax
2235                              <1>
2236 00009310 6689D8              <1>       mov    ax, bx
2237 00009313 66C1E805            <1>       shr    ax, 5        ; shift right 5 times
2238 00009317 6683E00F            <1>       and    ax, 00001111b; Month Mask
2239 0000931B D40A                <1>       aam
2240 0000931D 660D3030            <1>       or     ax, '00'
2241 00009321 86E0                <1>       xchg   ah, al
2242 00009323 66A3[EC3B0100]      <1>       mov    [File_Month], ax
2243                              <1>
2244 00009329 6689D8              <1>       mov    ax, bx
2245 0000932C 66C1E809            <1>       shr     ax, 9
2246 00009330 6683E07F            <1>       and    ax, 01111111b; Result = Year - 1980
2247 00009334 6605BC07            <1>       add    ax, 1980
2248                              <1>
2249 00009338 B10A                <1>       mov    cl, 10
2250 0000933A F6F1                <1>       div    cl           ; Q -> AL, R -> AH
2251 0000933C 80CC30              <1>       or     ah, '0'
2252 0000933F 8825[F23B0100]      <1>       mov    [File_Year+3], ah
2253 00009345 D40A                <1>       aam
2254 00009347 86E0                <1>       xchg   ah, al
2255 00009349 80CC30              <1>       or     ah, '0'        ; Convert to ASCII
2256 0000934C 8825[F13B0100]      <1>       mov    [File_Year+2], ah
2257 00009352 D40A                <1>       aam
2258 00009354 86C4                <1>       xchg   al, ah
2259 00009356 660D3030            <1>       or     ax, '00'
2260 0000935A 66A3[EF3B0100]      <1>       mov    [File_Year], ax
2261                              <1>
2262                              <1> loc_show_line:
2263 00009360 56                  <1>       push   esi
2264 00009361 BE[CC3B0100]        <1>       mov     esi, File_Name
2265 00009366 E8DAE1FFFF          <1>       call   print_msg
2266 0000936B BE[BF440100]        <1>       mov    esi, nextline
2267 00009370 E8D0E1FFFF          <1>       call   print_msg
2268 00009375 5E                  <1>       pop    esi
```

```
2269                                    <1>
2270 00009376 FE0D[958B0100]            <1>       dec    byte [PrintDir_RowCounter]
2271 0000937C 0F84D4000000              <1>         jz      pause_dir_scroll
2272                                    <1>
2273                                    <1> loc_next_entry:
2274 00009382 E899010000                <1>       call   find_next_file
2275 00009387 0F8391FEFFFF              <1>         jnc     loc_dfname_use_this
2276                                    <1>
2277                                    <1> loc_dir_ok:
2278 0000938D B90A000000                <1>       mov     ecx, 10
2279 00009392 66A1[808B0100]            <1>       mov    ax, [Dir_Count]
2280 00009398 BF[0D3C0100]              <1>       mov    edi, Decimal_Dir_Count
2281 0000939D 6639C8                    <1>       cmp    ax, cx ; 10
2282 000093A0 7216                      <1>       jb     short pass_ddc
2283 000093A2 47                        <1>       inc    edi
2284 000093A3 6683F864                  <1>       cmp    ax, 100
2285 000093A7 720F                      <1>       jb     short pass_ddc
2286 000093A9 47                        <1>       inc    edi
2287 000093AA 663DE803                  <1>       cmp    ax, 1000
2288 000093AE 7208                      <1>       jb     short pass_ddc
2289 000093B0 47                        <1>       inc    edi
2290 000093B1 663D1027                  <1>       cmp    ax, 10000
2291 000093B5 7201                      <1>       jb     short pass_ddc
2292 000093B7 47                        <1>       inc    edi
2293                                    <1> pass_ddc:
2294 000093B8 886F01                    <1>       mov     [edi+1], ch ; 0
2295                                    <1> loc_ddc_rediv:
2296 000093BB 31D2                      <1>       xor     edx, edx
2297 000093BD 66F7F1                    <1>       div     cx   ; 10
2298 000093C0 80C230                    <1>       add     dl, '0'
2299 000093C3 8817                      <1>       mov     [edi], dl
2300 000093C5 4F                        <1>       dec     edi
2301 000093C6 6609C0                    <1>       or     ax, ax
2302 000093C9 75F0                      <1>       jnz    short loc_ddc_rediv
2303                                    <1>
2304 000093CB 66A1[7E8B0100]            <1>       mov     ax, [File_Count]
2305 000093D1 BF[FC3B0100]              <1>       mov     edi, Decimal_File_Count
2306 000093D6 6639C8                    <1>       cmp     ax, cx ; 10
2307 000093D9 7216                      <1>       jb      short pass_dfc
2308 000093DB 47                        <1>       inc     edi
2309 000093DC 6683F864                  <1>       cmp     ax, 100
2310 000093E0 720F                      <1>       jb      short pass_dfc
2311 000093E2 47                        <1>       inc     edi
2312 000093E3 663DE803                  <1>       cmp     ax, 1000
2313 000093E7 7208                      <1>       jb      short pass_dfc
2314 000093E9 47                        <1>       inc     edi
2315 000093EA 663D1027                  <1>       cmp     ax, 10000
2316 000093EE 7201                      <1>       jb      short pass_dfc
2317 000093F0 47                        <1>       inc     edi
2318                                    <1> pass_dfc:
2319                                    <1>       ;mov    cx, 10
2320 000093F1 886F01                    <1>       mov     [edi+1], ch ; 00
2321                                    <1> loc_dfc_rediv:
2322                                    <1>       ;xor   dx, dx
2323 000093F4 30D2                      <1>       xor    dl, dl
2324 000093F6 66F7F1                    <1>       div    cx
2325 000093F9 80C230                    <1>       add    dl, '0'
2326 000093FC 8817                      <1>       mov    [edi], dl
2327 000093FE 4F                        <1>       dec    edi
2328 000093FF 6609C0                    <1>       or     ax, ax
2329 00009402 75F0                      <1>       jnz    short loc_dfc_rediv
2330                                    <1>
2331 00009404 BF[948B0100]              <1>       mov     edi, TFS_Dec_End
2332                                    <1>         ;mov    byte [edi], 0
2333 00009409 A1[828B0100]              <1>       mov     eax, [Total_FSize]
2334                                    <1>       ;mov    ecx, 10
2335                                    <1> rediv_tfs_hex:
2336                                    <1>       ;sub   edx, edx
2337 0000940E 28D2                      <1>       sub    dl, dl
2338 00009410 F7F1                      <1>       div    ecx
2339 00009412 80C230                    <1>       add    dl, '0'
2340 00009415 4F                        <1>       dec     edi
2341 00009416 8817                      <1>       mov     [edi], dl
2342 00009418 21C0                      <1>       and    eax, eax
2343 0000941A 75F2                      <1>       jnz    short rediv_tfs_hex
2344                                    <1>
2345 0000941C 893D[868B0100]            <1>       mov    [TFS_Dec_Begin], edi
2346 00009422 BE[FA3B0100]              <1>       mov    esi, Decimal_File_Count_Header
2347 00009427 E819E1FFFF                <1>       call   print_msg
2348 0000942C BE[023C0100]              <1>       mov    esi, str_files
2349 00009431 E80FE1FFFF                <1>       call   print_msg
2350 00009436 BE[133C0100]              <1>       mov    esi, str_dirs
2351 0000943B E805E1FFFF                <1>       call   print_msg
2352 00009440 8B35[868B0100]            <1>       mov    esi, [TFS_Dec_Begin]
2353 00009446 E8FAE0FFFF                <1>       call   print_msg
2354 0000944B BE[243C0100]              <1>       mov    esi, str_bytes
2355 00009450 E8F0E0FFFF                <1>       call   print_msg
2356                                    <1>
2357 00009455 C3                        <1>       retn
2358                                    <1>
2359                                    <1> pause_dir_scroll:
2360 00009456 28E4                      <1>       sub    ah, ah
2361 00009458 E8887AFFFF                <1>       call   int16h
2362 0000945D 3C1B                      <1>       cmp    al, 1Bh
2363 0000945F 0F8428FFFFFF              <1>         je      loc_dir_ok
2364 00009465 C605[958B0100]10          <1>       mov    byte [PrintDir_RowCounter], 16 ; Reset counter
2365 0000946C 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
```

```
                                <1>       ; INPUT ->
                                <1>       ;     ESI = ASCIIZ File/Dir Name Address (in Current Directory)
                                <1>       ;     AL = Attributes AND mask (The AND result must be equal to AL)
                                <1>       ;           bit 0 = Read Only
                                <1>       ;           bir 1 = Hidden
                                <1>       ;           bit 2 = System
                                <1>       ;           bit 3 = Volume Label
                                <1>       ;           bit 4 = Directory
                                <1>       ;           bit 5 = Archive
                                <1>       ;           bit 6 = Reserved, must be 0
                                <1>       ;           bit 7 = Reserved, must be 0
                                <1>       ;      AH = Attributes Negative AND mask (The AND result must be ZERO)
                                <1>       ;
                                <1>       ; OUTPUT ->
                                <1>       ;     CF = 1 -> Error, Error Code in EAX (AL)
                                <1>       ;     CF = 0 ->
                                <1>       ;          ESI = Directory Entry (FindFile_DirEntry) Location
                                <1>       ;          EDI = Directory Buffer Directory Entry Location
                                <1>       ;          EAX = File Size
                                <1>       ;           BL = Attributes of The File/Directory
                                <1>       ;           BH = Long Name Yes/No Status (>0 is YES)
                                <1>       ;             DX > 0 : Ambiguous filename chars are used
                                <1>       ;
                                <1>       ; (EAX, EBX, ECX, EDX, ESI, EDI will be changed)
                                <1>
00009471 66A3[468B0100]         <1>       mov   [FindFile_AttributesMask], ax
00009477 BF[488B0100]           <1>       mov   edi, FindFile_DirEntry ; TR-DOS Fullfilename formatted buffer
0000947C 31C0                   <1>       xor   eax, eax
0000947E B90B000000             <1>       mov   ecx, 11
00009483 F3AB                   <1>       rep   stosd  ; 44 bytes
                                <1>       ;stosw       ; +2 bytes
                                <1>
00009485 BF[388B0100]           <1>       mov   edi, FindFile_Name ; FFF structure, offset 66
0000948A 39FE                   <1>       cmp   esi, edi
0000948C 7408                   <1>       je    short loc_fff_mfn_ok
0000948E 89FA                   <1>       mov   edx, edi
                                <1>       ; move 13 bytes
00009490 A5                     <1>       movsd
00009491 A5                     <1>       movsd
00009492 A5                     <1>       movsd
00009493 AA                     <1>       stosb
00009494 89D6                   <1>       mov   esi, edx
                                <1> loc_fff_mfn_ok:
00009496 BF[E78A0100]           <1>       mov   edi, Dir_Entry_Name ; Dir Entry Format File Name
0000949B E8D7200000             <1>       call  convert_file_name
000094A0 89FE                   <1>       mov   esi, edi ; offset Dir_Entry_Name
                                <1>
000094A2 66A1[468B0100]         <1>       mov   ax, [FindFile_AttributesMask]
                                <1>       ;xor  ecx, ecx
000094A8 30C9                   <1>       xor   cl, cl
000094AA E8D11D0000             <1>       call  locate_current_dir_file
000094AF 726E                   <1>       jc    short loc_fff_retn
                                <1>       ; EDI = Directory Entry
                                <1>       ; EBX = Directory Buffer Entry Index/Number
                                <1>
                                <1> loc_fff_fnf_ln_check:
000094B1 30ED                   <1>       xor   ch, ch
000094B3 80F60F                 <1>       xor   dh, 0Fh
000094B6 7408                   <1>       jz    short loc_fff_longname_yes
000094B8 882D[458B0100]         <1>       mov   [FindFile_LongNameYes], ch ; 0
000094BE EB0C                   <1>       jmp   short loc_fff_longname_no
                                <1>
                                <1> loc_fff_longname_yes:
                                <1>       ;inc  byte [FindFile_LongNameYes]
000094C0 8A0D[528A0100]         <1>       mov   cl, [LFN_EntryLength]
000094C6 880D[458B0100]         <1>       mov   [FindFile_LongNameEntryLength], cl ; FindFile_LongNameYes
                                <1>
                                <1> loc_fff_longname_no:
                                <1>       ;mov  bx, [DirBuff_CurrentEntry]
000094CC 66891D[708B0100]       <1>       mov   [FindFile_DirEntryNumber], bx
000094D3 6689C2                 <1>       mov   dx, ax ; Ambiguous Filename chars used sign > 0
                                <1>
000094D6 A0[56820100]           <1>       mov   al, [Current_Drv]
000094DB A2[F68A0100]           <1>       mov   [FindFile_Drv], al
                                <1>
000094E0 A1[50820100]           <1>       mov   eax, [Current_Dir_FCluster]
000094E5 A3[688B0100]           <1>       mov   [FindFile_DirFirstCluster], eax
                                <1>
000094EA A1[81890100]           <1>       mov   eax, [DirBuff_Cluster]
000094EF A3[6C8B0100]           <1>       mov   [FindFile_DirCluster], eax
                                <1>
000094F4 66FF05[728B0100]       <1>       inc   word [FindFile_MatchCounter]
                                <1>
000094FB 89FB                   <1>       mov   ebx, edi
000094FD 89FE                   <1>       mov   esi, edi
000094FF BF[488B0100]           <1>       mov   edi, FindFile_DirEntry
00009504 89F8                   <1>       mov   eax, edi
00009506 B108                   <1>       mov   cl, 8
00009508 F3A5                   <1>       rep   movsd
0000950A 89C6                   <1>       mov   esi, eax
0000950C 89DF                   <1>       mov   edi, ebx
                                <1>
0000950E A1[648B0100]           <1>       mov   eax, [FindFile_DirEntry+28] ; File Size
                                <1>
00009513 8A1D[538B0100]         <1>       mov   bl, [FindFile_DirEntry+11] ; File Attributes
00009519 8A3D[458B0100]         <1>       mov   bh, [FindFile_LongNameYes]
                                <1>
                                <1>       ;mov  cx, [DirBuff_EntryCounter]
                                <1>       ;mov  [FindFile_DirEntryNumber], cx
                                <1>       ;mov  cx, [FindFile_DirEntryNumber]
                                <1>       ; ecx = 0
                                <1>
                                <1> loc_fff_retn:
0000951F C3                     <1>       retn
                                <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 00009520 66833D[728B0100]00  <1>       cmp    word [FindFile_MatchCounter], 0
2501 00009528 7707                <1>       ja     short loc_start_search_next_file
2502                                <1>
2503                                <1> loc_fnf_stc_retn:
2504 0000952A F9                  <1>       stc
2505                                <1> loc_fnf_ax12h_retn:
2506 0000952B B80C000000          <1>       mov    eax, 12 ; No More files
2507                                <1> ;loc_fnf_retn:
2508 00009530 C3                  <1>       retn
2509                                <1>
2510                                <1> loc_start_search_next_file:
2511 00009531 668B1D[708B0100]    <1>       mov    bx, [FindFile_DirEntryNumber]
2512 00009538 6643                <1>       inc    bx
2513 0000953A 663B1D[7F890100]    <1>       cmp    bx, [DirBuff_LastEntry]
2514 00009541 7719                <1>       ja     short loc_cont_search_next_file
2515                                <1>
2516                                <1> loc_fnf_search:
2517 00009543 BE[E78A0100]        <1>       mov    esi, Dir_Entry_Name
2518 00009548 66A1[468B0100]      <1>       mov    ax, [FindFile_AttributesMask]
2519 0000954E 6631C9              <1>       xor    cx, cx
2520 00009551 E82E1E0000          <1>       call   find_directory_entry
2521 00009556 0F8355FFFFFF        <1>         jnc     loc_fff_fnf_ln_check
2522                                <1>
2523                                <1> loc_cont_search_next_file:
2524 0000955C 31DB                <1>       xor    ebx, ebx
2525 0000955E 8A3D[56820100]      <1>       mov    bh, [Current_Drv]
2526 00009564 BE00010900          <1>       mov    esi, Logical_DOSDisks
2527 00009569 01DE                <1>       add    esi, ebx
2528                                <1>
2529 0000956B 803D[54820100]00    <1>       cmp    byte [Current_Dir_Level], 0
2530 00009572 7608                <1>       jna    short loc_fnf_check_FAT_type
2531 00009574 807E0301            <1>       cmp    byte [esi+LD_FATType], 1
2532 00009578 72B1                <1>       jb     short loc_fnf_ax12h_retn
2533 0000957A EB06                <1>       jmp    short loc_fnf_check_next_cluster
2534                                <1>
2535                                <1> loc_fnf_check_FAT_type:
2536 0000957C 807E0303            <1>       cmp    byte [esi+LD_FATType], 3
2537 00009580 72A9                <1>       jb     short loc_fnf_ax12h_retn
2538                                <1>
2539                                <1> loc_fnf_check_next_cluster:
2540 00009582 A1[81890100]        <1>       mov    eax, [DirBuff_Cluster]
2541 00009587 E8CA370000          <1>       call   get_next_cluster
2542 0000958C 7306                <1>       jnc    short loc_fnf_load_next_dir_cluster
2543 0000958E 09C0                <1>       or     eax, eax
2544 00009590 7498                <1>       jz     short loc_fnf_stc_retn
2545                                <1>       ;mov   eax, 17 ;Drive not ready or read error
2546 00009592 F5                  <1>       cmc    ;stc
2547                                <1> loc_fnf_retn:
2548 00009593 C3                  <1>       retn
2549                                <1>
2550                                <1> loc_fnf_load_next_dir_cluster:
2551 00009594 E8A3390000          <1>       call   load_FAT_sub_directory
2552 00009599 72F8                <1>       jc     short loc_fnf_retn
2553 0000959B 6631DB              <1>       xor    bx, bx
2554 0000959E 66891D[708B0100]    <1>       mov    [FindFile_DirEntryNumber], bx
2555 000095A5 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 000095A7 803E20              <1>       cmp    byte [esi], 20h
2564 000095AA 7701                <1>       ja     short loc_find_longname
2565                                <1>       ;jb    short loc_longname_retn
2566                                <1>       ;inc   esi
2567                                <1>       ;je    short get_longname_fchar
2568                                <1> ;loc_longname_retn:
2569 000095AC C3                  <1>       retn
2570                                <1> loc_find_longname:
2571 000095AD E839210000          <1>       call   find_longname
2572 000095B2 7328                <1>       jnc    short loc_print_longname
2573                                <1>
2574 000095B4 08C0                <1>       or     al, al
2575 000095B6 741A                <1>       jz     short loc_longname_not_found
2576                                <1>
2577                                <1>       ; 16/10/2016 (15h -> 15, 17)
2578 000095B8 3C0F                <1>       cmp    al, 15
2579 000095BA 0F84B6F7FFFF        <1>       je     cd_drive_not_ready ; drive not ready
2580                                <1>                                 ; or
2581 000095C0 3C11                <1>       cmp    al, 17              ; read error
2582 000095C2 0F84AEF7FFFF        <1>       je     cd_drive_not_ready
2583                                <1>
```

```
2584                                <1> loc_ln_file_dir_not_found:
2585 000095C8 BE[4E3B0100]        <1>       mov    esi, Msg_File_Directory_Not_Found
2586                                <1>       ;call  print_msg
2587                                <1>         ;retn
2588 000095CD E973DFFFFF          <1>       jmp    print_msg
2589                                <1>
2590                                <1> loc_longname_not_found:
2591 000095D2 BE[6D3B0100]        <1>         mov     esi, Msg_LongName_Not_Found
2592                                <1>       ;call  print_msg
2593                                <1>         ;retn
2594 000095D7 E969DFFFFF          <1>       jmp    print_msg
2595                                <1>
2596                                <1> loc_print_longname:
2597                                <1>       ;mov   esi, LongFileName
2598 000095DC BF[56830100]        <1>       mov    edi, TextBuffer
2599 000095E1 57                  <1>       push   edi
2600 000095E2 3C00                <1>       cmp    al, 0
2601 000095E4 7708                <1>       ja     short loc_print_longname_1
2602                                <1> loc_print_FS_longname: ; Singlix FS (64 byte ASCIIZ file name)
2603 000095E6 AC                  <1>       lodsb
2604 000095E7 AA                  <1>       stosb
2605 000095E8 08C0                <1>       or     al, al
2606 000095EA 75FA                <1>       jnz    short loc_print_FS_longname
2607 000095EC EB07                <1>       jmp    short loc_print_longname_2
2608                                <1>       ;
2609                                <1> loc_print_longname_1: ; MS Windows long name (UNICODE chars)
2610 000095EE 66AD                <1>       lodsw
2611 000095F0 AA                  <1>       stosb
2612 000095F1 08C0                <1>       or     al, al
2613 000095F3 75F9                <1>       jnz    short loc_print_longname_1
2614                                <1>       ;
2615                                <1> loc_print_longname_2:
2616 000095F5 5E                  <1>       pop    esi
2617 000095F6 E84ADFFFFF          <1>       call   print_msg
2618 000095FB BE[BF440100]        <1>       mov    esi, nextline
2619                                <1>       ;call  print_msg
2620                                <1>       ;retn
2621 00009600 E940DFFFFF          <1>       jmp    print_msg
2622                                <1>
2623                                <1> show_file:
2624                                <1>       ; 18/02/2016
2625                                <1>       ; 17/02/2016
2626                                <1>       ; 15/02/2016 (TRDOS 386 = TRDOS v2.0)
2627                                <1>       ; 13/09/2011 (CMD_INTR.ASM, 'cmp_cmd_show')
2628                                <1>       ; 08/11/2009
2629                                <1>
2630                                <1> loc_show_parse_path_name:
2631 00009605 BF[F68A0100]        <1>       mov    edi, FindFile_Drv
2632 0000960A E833200000          <1>       call   parse_path_name
2633 0000960F 0F824CF9FFFF        <1>       jc     loc_cmd_failed
2634                                <1>
2635                                <1> loc_show_check_filename_exists:
2636 00009615 BE[388B0100]        <1>       mov    esi, FindFile_Name
2637 0000961A 803E20              <1>       cmp    byte [esi], 20h
2638 0000961D 0F863EF9FFFF        <1>       jna    loc_cmd_failed
2639                                <1>
2640                                <1>       ; 15/02/2016 (invalid file name check)
2641 00009623 E807020000          <1>       call   check_filename
2642 00009628 730A                <1>       jnc    short loc_show_change_drv
2643                                <1>
2644 0000962A BE[3A3C0100]        <1>       mov    esi, Msg_invalid_name_chars
2645 0000962F E911DFFFFF          <1>       jmp    print_msg
2646                                <1>
2647                                <1> loc_show_change_drv:
2648 00009634 8A35[56820100]      <1>       mov    dh, [Current_Drv]
2649 0000963A 8835[B2890100]      <1>       mov    [RUN_CDRV], dh
2650 00009640 8A15[F68A0100]      <1>       mov    dl, [FindFile_Drv]
2651 00009646 38F2                <1>       cmp    dl, dh
2652 00009648 740B                <1>       je     short loc_show_change_directory
2653 0000964A E87FEAFFFF          <1>       call   change_current_drive
2654                                <1>       ;jc    loc_file_rw_cmd_failed
2655 0000964F 0F8237F9FFFF        <1>       jc     loc_run_cmd_failed
2656                                <1>
2657                                <1> loc_show_change_directory:
2658 00009655 803D[F78A0100]20    <1>       cmp    byte [FindFile_Directory], 20h
2659 0000965C 7618                <1>       jna    short loc_findload_showfile
2660                                <1>
2661 0000965E FE05[23380100]      <1>       inc    byte [Restore_CDIR]
2662 00009664 BE[F78A0100]        <1>       mov    esi, FindFile_Directory
2663 00009669 30E4                <1>       xor    ah, ah ; CD_COMMAND sign -> 0
2664 0000966B E8BC190000          <1>       call   change_current_directory
2665                                <1>       ;jc    loc_file_rw_cmd_failed
2666 00009670 0F8216F9FFFF        <1>       jc     loc_run_cmd_failed
2667                                <1>
2668                                <1> ;loc_show_change_prompt_dir_string:
2669                                <1>       ;call  change_prompt_dir_string
2670                                <1>
2671                                <1> loc_findload_showfile:
2672                                <1>       ; 15/02/2016
2673 00009676 BE[388B0100]        <1>       mov    esi, FindFile_Name
2674 0000967B BF[E78A0100]        <1>       mov    edi, Dir_Entry_Name ; Dir Entry Format File Name
2675 00009680 E8F21E0000          <1>       call   convert_file_name
2676 00009685 89FE                <1>       mov    esi, edi ; offset Dir_Entry_Name
2677                                <1>
2678 00009687 28C0                <1>       sub    al, al ; Attrib AND mask = 0
2679                                <1>       ; Directory attribute : 10h
2680                                <1>       ; Volume name attribute: 8h
2681 00009689 B418                <1>       mov    ah, 00011000b ; 18h (Attrib NAND, AND --> zero mask)
2682                                <1>       ;
2683 0000968B 6631C9              <1>       xor    cx, cx
2684 0000968E E8ED1B0000          <1>       call   locate_current_dir_file
2685                                <1>       ;jc    loc_file_rw_cmd_failed
2686 00009693 0F82F3F8FFFF        <1>       jc     loc_run_cmd_failed
2687                                <1>
2688                                <1> loc_show_load_file:
```

```
2689                                    <1>       ; EDI = Directory Entry
2690 00009699 668B4714                 <1>       mov    ax, [edi+DirEntry_FstClusHI] ; First Cluster High Word
2691 0000969D C1E010                   <1>       shl    eax, 16
2692 000096A0 668B471A                 <1>       mov    ax, [edi+DirEntry_FstClusLO] ; First Cluster Low Word
2693 000096A4 A3[A08B0100]             <1>       mov    [Show_Cluster], eax
2694 000096A9 8B471C                   <1>       mov    eax, [edi+DirEntry_FileSize] ; File Size
2695 000096AC 21C0                     <1>       and    eax, eax ; Empty file !
2696 000096AE 0F8491000000             <1>         jz     end_of_show_file
2697 000096B4 A3[A48B0100]             <1>       mov    [Show_FileSize], eax
2698 000096B9 31C0                     <1>       xor    eax, eax
2699 000096BB A3[A88B0100]             <1>       mov    [Show_FilePointer], eax ; 0
2700 000096C0 66A3[AC8B0100]           <1>       mov    [Show_ClusterPointer], ax ; 0
2701 000096C6 29DB                     <1>       sub    ebx, ebx
2702 000096C8 8A3D[56820100]           <1>       mov    bh, [Current_Drv]
2703 000096CE BE00010900               <1>       mov    esi, Logical_DOSDisks
2704 000096D3 01DE                     <1>       add    esi, ebx
2705 000096D5 8935[9C8B0100]           <1>       mov    [Show_LDDDT], esi ; Logical DOS Drv Description Table addr
2706                                    <1>
2707 000096DB 807E0300                 <1>       cmp    byte [esi+LD_FATType], 0
2708 000096DF 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 000096E1 8B15[A08B0100]           <1>       mov    edx, [Show_Cluster] ; Compatibility dir. buffer value (FDT)
2712 000096E7 8915[988B0100]           <1>       mov    [Show_FDT], edx
2713 000096ED 31C0                     <1>       xor    eax, eax
2714 000096EF A3[A08B0100]             <1>       mov    [Show_Cluster], eax ; Sector index  = 0
2715                                    <1>                                  ; (next time it will be 1)
2716                                    <1> loc_show_calculate_cluster_size:
2717 000096F4 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 000096F8 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 000096FB 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 000096FD 66A3[AE8B0100]           <1>       mov    [Show_ClusterSize], ax
2726                                    <1>
2727                                    <1> loc_start_show_file:
2728 00009703 BE[BF440100]             <1>       mov    esi, nextline
2729 00009708 E838DEFFFF               <1>       call   print_msg
2730                                    <1>
2731 0000970D A1[A08B0100]             <1>       mov    eax, [Show_Cluster]
2732 00009712 C605[B08B0100]17         <1>       mov    byte [Show_RowCount], 23
2733                                    <1>
2734                                    <1>       ; 17/02/2016
2735 00009719 8B35[9C8B0100]           <1>       mov    esi, [Show_LDDDT]
2736                                    <1>
2737                                    <1> loc_show_next_cluster:
2738                                    <1>       ; 15/02/2016
2739 0000971F BB00000700               <1>       mov    ebx, Cluster_Buffer ; 70000h (for current TRDOS 386 version)
2740                                    <1>       ; ESI = Logical DOS drv description table address
2741 00009724 E851380000               <1>       call   read_cluster
2742                                    <1>       ;jc    loc_file_rw_cmd_failed
2743 00009729 0F825DF8FFFF             <1>       jc     loc_run_cmd_failed
2744                                    <1>
2745 0000972F 31DB                     <1>       xor    ebx, ebx
2746                                    <1> loc_show_next_byte:
2747 00009731 803D[B08B0100]00         <1>       cmp    byte [Show_RowCount], 0
2748 00009738 7521                     <1>       jne    short pass_show_wait_for_key
2749 0000973A 30E4                     <1>       xor    ah, ah
2750 0000973C E8A477FFFF               <1>       call   int16h
2751 00009741 3C1B                     <1>       cmp    al, 1Bh
2752 00009743 750F                     <1>       jne    short pass_exit_show
2753                                    <1> end_of_show_file:
2754                                    <1> pass_show_file:
2755 00009745 BE[BF440100]             <1>       mov    esi, nextline
2756 0000974A E8F6DDFFFF               <1>       call   print_msg
2757 0000974F E94B010000               <1>       jmp    loc_file_rw_restore_retn
2758                                    <1>
2759                                    <1> pass_exit_show:
2760 00009754 C605[B08B0100]14         <1>       mov    byte [Show_RowCount], 20
2761                                    <1> pass_show_wait_for_key:
2762 0000975B 81C300000700             <1>       add    ebx, Cluster_Buffer
2763 00009761 8A03                     <1>       mov    al, [ebx]
2764 00009763 3C0D                     <1>       cmp    al, 0Dh
2765 00009765 0F8590000000             <1>         jne    loc_show_check_tab_space
2766 0000976B FE0D[B08B0100]           <1>       dec    byte [Show_RowCount]
2767                                    <1> pass_show_dec_rowcount:
2768 00009771 B307                     <1>       mov    bl, 7 ; (light gray character color, black background)
2769 00009773 8A3D[BE810100]           <1>       mov    bh, [ACTIVE_PAGE] ; [ptty]
2770 00009779 E87F8BFFFF               <1>       call   _write_tty
2771                                    <1> loc_show_check_eof:
2772 0000977E FF05[A88B0100]           <1>       inc    dword [Show_FilePointer]
2773 00009784 A1[A88B0100]             <1>       mov    eax, [Show_FilePointer]
2774 00009789 3B05[A48B0100]           <1>       cmp    eax, [Show_FileSize]
2775 0000978F 73B4                     <1>       jnb    short end_of_show_file
2776 00009791 66FF05[AC8B0100]         <1>       inc    word [Show_ClusterPointer]
2777 00009798 0FB71D[AC8B0100]         <1>       movzx  ebx, word [Show_ClusterPointer]
2778                                    <1>
2779                                    <1>       ; 17/02/2016
2780                                    <1>       ; (sector boundary -9 bits- check, 512 = 0)
2781 0000979F 66F7C3FF01               <1>         test   bx, 1FFh ;  1 to 511
2782 000097A4 758B                     <1>       jnz    short loc_show_next_byte
2783                                    <1>
2784                                    <1>       ; 16/02/2016
2785 000097A6 8B35[9C8B0100]           <1>       mov    esi, [Show_LDDDT]
2786                                    <1>       ;
2787 000097AC 807E0300                 <1>       cmp    byte [esi+LD_FATType], 0
2788 000097B0 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 000097B2 A1[A08B0100]             <1>       mov    eax, [Show_Cluster]
2793 000097B7 40                       <1>       inc    eax
```

```
2794 000097B8 A3[A08B0100]        <1>       mov    [Show_Cluster], eax
2795                              <1>
2796 000097BD 6621DB              <1>       and    bx, bx ; 65536 -> 0
2797 000097C0 0F856BFFFFFF        <1>         jnz  loc_show_next_byte
2798 000097C6 E954FFFFFF          <1>       jmp     loc_show_next_cluster
2799                              <1>
2800                              <1> loc_show_check_fat_cluster_size:
2801                              <1>       ; 17/02/2016
2802 000097CB 663B1D[AE8B0100]    <1>       cmp    bx, [Show_ClusterSize] ; cluster size in bytes
2803 000097D2 0F8259FFFFFF        <1>         jb   loc_show_next_byte
2804 000097D8 66C705[AC8B0100]00- <1>       mov    word [Show_ClusterPointer], 0
2804 000097E0 00                  <1>
2805                              <1>
2806 000097E1 A1[A08B0100]        <1>       mov    eax, [Show_Cluster]
2807                              <1>       ;mov   esi, [Show_LDDDT]
2808                              <1> loc_show_get_next_cluster:
2809 000097E6 E86B350000          <1>       call   get_next_cluster
2810                              <1>       ;jc    loc_file_rw_cmd_failed
2811 000097EB 0F829BF7FFFF        <1>       jc     loc_run_cmd_failed
2812                              <1> loc_show_update_ccluster:
2813 000097F1 A3[A08B0100]        <1>       mov    [Show_Cluster], eax
2814 000097F6 E924FFFFFF          <1>         jmp     loc_show_next_cluster
2815                              <1>
2816                              <1> loc_show_check_tab_space:
2817 000097FB 3C09                <1>       cmp    al, 09h
2818 000097FD 0F856EFFFFFF        <1>         jne     pass_show_dec_rowcount
2819                              <1> loc_show_put_tab_space:
2820 00009803 8A3D[BE810100]      <1>       mov    bh, [ACTIVE_PAGE] ; [ptty]
2821 00009809 E86B87FFFF          <1>       call   get_cpos
2822                              <1>       ; dl = cursor column
2823 0000980E 80E207              <1>       and    dl, 7 ; 18/02/2016
2824                              <1>       ;shr   bh, 1 ; [ACTIVE_PAGE]
2825 00009811 8A3D[BE810100]      <1>       mov    bh, [ACTIVE_PAGE]
2826 00009817 B307                <1>       mov    bl, 7 ; color attribute
2827                              <1> loc_show_put_space_chars:
2828 00009819 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 0000981B 52                  <1>       push   edx ; 29/12/2017
2833 0000981C E8DC8AFFFF          <1>       call   _write_tty
2834 00009821 5A                  <1>       pop    edx ; 29/12/2017
2835                              <1>       ;pop   dx
2836                              <1>       ; 18/02/2016
2837 00009822 80FA07              <1>       cmp    dl, 7
2838 00009825 0F8353FFFFFF        <1>       jnb    loc_show_check_eof
2839 0000982B FEC2                <1>       inc    dl
2840 0000982D 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 0000982F 56                  <1>       push   esi
2876                              <1>
2877 00009830 BF[22390100]        <1>         mov     edi, invalid_fname_chars
2878 00009835 AC                  <1>       lodsb
2879                              <1> check_filename_next_char:
2880 00009836 B914000000          <1>       mov    ecx, sizeInvFnChars
2881 0000983B BF[22390100]        <1>       mov    edi, invalid_fname_chars
2882                              <1> loc_scan_invalid_filename_char:
2883 00009840 AE                  <1>       scasb
2884 00009841 741F                <1>       je     short loc_invalid_filename_stc
2885 00009843 E2FB                <1>       loop   loc_scan_invalid_filename_char
2886 00009845 AC                  <1>       lodsb
2887 00009846 3C1F                <1>       cmp    al, 1Fh  ; 20h and above
2888 00009848 77EC                <1>       ja     short check_filename_next_char
2889                              <1>
2890                              <1> check_filename_dot:
2891 0000984A 8B3424              <1>       mov    esi, [esp]
2892                              <1>
2893 0000984D B421                <1>       mov    ah, 21h
2894 0000984F B908000000          <1>       mov    ecx, 8
2895                              <1> loc_check_filename_next_char:
2896 00009854 AC                  <1>       lodsb
2897 00009855 3C2E                <1>       cmp    al, 2Eh
```

```
2898 00009857 7511              <1>       jne    short pass_check_fn_dot_check
2899                            <1> loc_check_filename_ext_0:
2900 00009859 AC                <1>       lodsb
2901 0000985A 38E0              <1>       cmp    al, ah ; 21h
2902 0000985C 7205              <1>       jb     short loc_invalid_filename
2903 0000985E 3C2E              <1>       cmp    al, 2Eh
2904 00009860 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 00009862 F9                <1>       stc
2909                            <1> loc_invalid_filename:
2910                            <1>       ; 10/10/2016 (0Bh -> 26)
2911 00009863 B81A000000        <1>       mov    eax, ERR_INV_FILE_NAME ; (=26)
2912                            <1>       ; Invalid file name chars
2913                            <1> loc_check_fn_rtn:
2914 00009868 5E                <1>       pop    esi
2915 00009869 C3                <1>       retn
2916                            <1>
2917                            <1> pass_check_fn_dot_check:
2918 0000986A 38E0              <1>       cmp    al, ah ; 21h
2919 0000986C 7224              <1>       jb     short loc_check_fn_clc_rtn
2920 0000986E E2E4              <1>       loop   loc_check_filename_next_char
2921 00009870 AC                <1>       lodsb
2922 00009871 38E0              <1>       cmp    al, ah ; 21h
2923 00009873 721D              <1>       jb     short loc_check_fn_clc_rtn
2924 00009875 3C2E              <1>       cmp    al, 2Eh
2925 00009877 75E9              <1>       jne    short loc_check_fn_stc_rtn
2926 00009879 EBDE              <1>       jmp    short loc_check_filename_ext_0
2927                            <1>
2928                            <1> loc_check_filename_ext_1:
2929 0000987B AC                <1>       lodsb
2930 0000987C 38E0              <1>       cmp    al, ah ; 21h
2931 0000987E 7212              <1>       jb     short loc_check_fn_clc_rtn
2932 00009880 3C2E              <1>       cmp    al, 2Eh
2933 00009882 74DE              <1>       je     short loc_check_fn_stc_rtn
2934 00009884 AC                <1>       lodsb
2935 00009885 38E0              <1>       cmp    al, ah ; 21h
2936 00009887 7209              <1>       jb     short loc_check_fn_clc_rtn
2937 00009889 3C2E              <1>       cmp    al, 2Eh
2938 0000988B 74D5              <1>       je     short loc_check_fn_stc_rtn
2939 0000988D AC                <1>       lodsb
2940 0000988E 38E0              <1>       cmp    al, ah ; 21h
2941 00009890 73D0              <1>       jnb    short loc_check_fn_stc_rtn
2942                            <1>
2943                            <1> loc_check_fn_clc_rtn:
2944 00009892 5E                <1>       pop    esi
2945 00009893 F8                <1>       clc
2946 00009894 C3                <1>       retn
2947                            <1>
2948                            <1> loc_print_deleted_message:
2949 00009895 BE[0F3D0100]      <1>       mov    esi, Msg_Deleted
2950 0000989A 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 0000989F 9C                <1>       pushf
2959 000098A0 E84FF7FFFF        <1>       call   restore_cdir_after_cmd_fail
2960 000098A5 9D                <1>       popf
2961 000098A6 720D              <1>       jc     short loc_file_rw_check_write_fault
2962 000098A8 C3                <1>       retn
2963                            <1>
2964                            <1> loc_permission_denied:
2965                            <1>       ; 27/02/2016
2966 000098A9 BE[1C3D0100]      <1>       mov    esi, Msg_Permission_Denied
2967 000098AE E892DCFFFF        <1>       call   print_msg
2968 000098B3 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 000098B5 3C12              <1>         cmp  al, 18 ; 05/11/2016
2973 000098B7 0F85D4F6FFFF      <1>       jne     loc_run_cmd_failed_cmp_al
2974 000098BD BE[033B0100]      <1>       mov    esi, Msg_Not_Ready_Write_Err
2975                            <1>       ;call  print_msg
2976                            <1>       ;retn
2977 000098C2 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>       ;
2986                            <1> get_mkdir_fchar:
2987                            <1>       ; esi = directory name
2988 000098C7 803E20            <1>       cmp    byte [esi], 20h
2989 000098CA 7701              <1>         ja   short loc_mkdir_parse_path_name
2990                            <1>
2991                            <1> loc_mkdir_nodirname_retn:
2992 000098CC C3                <1>       retn
2993                            <1>
2994                            <1> loc_mkdir_parse_path_name:
2995 000098CD BF[F68A0100]      <1>       mov    edi, FindFile_Drv
2996 000098D2 E86B1D0000        <1>         call    parse_path_name
2997 000098D7 0F8284F6FFFF      <1>       jc     loc_cmd_failed
2998                            <1>
2999                            <1> loc_mkdir_check_dirname_exists:
3000 000098DD BE[388B0100]      <1>       mov    esi, FindFile_Name
3001 000098E2 803E20            <1>       cmp    byte [esi], 20h
3002 000098E5 0F8676F6FFFF      <1>       jna    loc_cmd_failed
```

```
3003 000098EB 8935[B48B0100]      <1>       mov   [DelFile_FNPointer], esi
3004 000098F1 E839FFFFFF          <1>       call  check_filename
3005 000098F6 7259                <1>       jc    short loc_mkdir_invalid_dir_name_chars
3006                              <1>
3007                              <1> loc_mkdir_drv:
3008 000098F8 8A35[56820100]      <1>       mov   dh, [Current_Drv]
3009 000098FE 8835[B2890100]      <1>       mov   [RUN_CDRV], dh
3010                              <1>
3011 00009904 8A15[F68A0100]      <1>       mov   dl, [FindFile_Drv]
3012 0000990A 38F2                <1>       cmp   dl, dh
3013 0000990C 7407                <1>       je    short loc_mkdir_change_directory
3014                              <1>
3015 0000990E E8BBE7FFFF          <1>       call  change_current_drive
3016 00009913 728A                <1>       jc    loc_file_rw_cmd_failed
3017                              <1>
3018                              <1> loc_mkdir_change_directory:
3019 00009915 803D[F78A0100]20    <1>       cmp   byte [FindFile_Directory], 20h
3020 0000991C 7614                <1>       jna   short loc_mkdir_find_directory
3021                              <1>
3022 0000991E FE05[23380100]      <1>       inc   byte [Restore_CDIR]
3023 00009924 BE[F78A0100]        <1>       mov   esi, FindFile_Directory
3024 00009929 30E4                <1>       xor   ah, ah ; CD_COMMAND sign -> 0
3025 0000992B E8FC160000          <1>       call  change_current_directory
3026 00009930 722E                <1>       jc    short loc_mkdir_check_error_code
3027                              <1>
3028                              <1> ;loc_mkdir_change_prompt_dir_string:
3029                              <1>       ;call  change_prompt_dir_string
3030                              <1>
3031                              <1> loc_mkdir_find_directory:
3032                              <1>       ;mov   esi, FindFile_Name
3033 00009932 8B35[B48B0100]      <1>       mov   esi, [DelFile_FNPointer]
3034                              <1>       ;xor   eax, eax
3035 00009938 6631C0              <1>       xor   ax, ax ; any name (dir, file, volume)
3036 0000993B E831FBFFFF          <1>       call  find_first_file
3037 00009940 721E                <1>       jc    short loc_mkdir_check_error_code
3038                              <1>
3039                              <1> loc_mkdir_directory_found:
3040 00009942 BE[673C0100]        <1>       mov   esi, Msg_Name_Exists
3041 00009947 E8F9DBFFFF          <1>       call  print_msg
3042                              <1>
3043 0000994C E94EFFFFFF          <1>         jmp     loc_file_rw_restore_retn
3044                              <1>
3045                              <1> loc_mkdir_invalid_dir_name_chars:
3046 00009951 BE[3A3C0100]        <1>       mov   esi, Msg_invalid_name_chars
3047 00009956 E8EADBFFFF          <1>       call  print_msg
3048                              <1>
3049 0000995B E93FFFFFFF          <1>         jmp     loc_file_rw_restore_retn
3050                              <1>
3051                              <1> loc_mkdir_check_error_code:
3052 00009960 3C02                <1>       cmp   al, 2
3053                              <1>       ;je   short loc_mkdir_directory_not_found
3054 00009962 7406                <1>       je    short loc_mkdir_ask_for_yes_no
3055 00009964 F9                  <1>       stc
3056 00009965 E935FFFFFF          <1>         jmp     loc_file_rw_cmd_failed
3057                              <1>
3058                              <1> loc_mkdir_directory_not_found:
3059                              <1> loc_mkdir_ask_for_yes_no:
3060 0000996A BE[883C0100]        <1>       mov   esi, Msg_DoYouWantMkdir
3061 0000996F E8D1DBFFFF          <1>       call  print_msg
3062 00009974 8B35[B48B0100]      <1>       mov   esi, [DelFile_FNPointer]
3063 0000997A E8C6DBFFFF          <1>       call  print_msg
3064 0000997F BE[A73C0100]        <1>       mov   esi, Msg_YesNo
3065 00009984 E8BCDBFFFF          <1>       call  print_msg
3066                              <1>
3067 00009989 C605[B13C0100]20    <1>       mov   byte [Y_N_nextline], 20h
3068                              <1>
3069                              <1> loc_mkdir_ask_again:
3070 00009990 30E4                <1>       xor   ah, ah
3071 00009992 E84E75FFFF          <1>       call  int16h
3072 00009997 3C1B                <1>       cmp   al, 1Bh
3073                              <1>       ;je   short loc_do_not_make_directory
3074 00009999 7439                <1>       je    short loc_mkdir_y_n_escape
3075 0000999B 24DF                <1>       and   al, 0DFh ; y -> Y, n -> N
3076 0000999D 3C59                <1>       cmp   al, 'Y' ; 'yes'
3077 0000999F 7404                <1>       je    short loc_mkdir_yes_make_directory
3078 000099A1 3C4E                <1>       cmp   al, 'N' ; 'no'
3079 000099A3 75EB                <1>       jne   short loc_mkdir_ask_again
3080                              <1>
3081                              <1> loc_do_not_make_directory:
3082                              <1> loc_mkdir_yes_make_directory:
3083 000099A5 E82E000000          <1>       call  y_n_answer ; 29/12/2017
3084                              <1>       ;cmp  al, 'Y' ; 'yes'
3085                              <1>       ;cmc
3086                              <1>         ;jnc loc_file_rw_restore_retn
3087 000099AA 3C4E                <1>       cmp   al, 'N' ; 'no'
3088 000099AC 0F84EDFEFFFF        <1>         je  loc_file_rw_restore_retn
3089                              <1>
3090                              <1> loc_mkdir_call_make_sub_directory:
3091 000099B2 8B35[B48B0100]      <1>       mov   esi, [DelFile_FNPointer]
3092 000099B8 B110                <1>       mov   cl, 10h ; Directory attributes
3093 000099BA E8821D0000          <1>       call  make_sub_directory
3094                              <1> loc_rename_file_ok: ; 06/03/2016
3095 000099BF 0F82DAFEFFFF        <1>         jc  loc_file_rw_cmd_failed
3096                              <1> move_source_file_to_destination_OK:
3097 000099C5 BE[B53C0100]        <1>       mov   esi, Msg_OK
3098 000099CA E876DBFFFF          <1>       call  print_msg
3099 000099CF E9CBFEFFFF          <1>       jmp   loc_file_rw_restore_retn
3100                              <1>
3101                              <1> loc_mkdir_y_n_escape:
3102 000099D4 B04E                <1>       mov   al, 'N' ; 'no'
3103 000099D6 EBCD                <1>       jmp   short loc_do_not_make_directory
3104                              <1>
3105                              <1> y_n_answer:
3106                              <1>       ; 29/12/2017
3107 000099D8 A2[B13C0100]        <1>       mov   [Y_N_nextline], al
```

```
3108                                    <1>       ;push  ax
3109 000099DD 50                        <1>       push   eax
3110 000099DE BE[B13C0100]              <1>       mov    esi, Y_N_nextline
3111 000099E3 E85DDBFFFF                <1>       call   print_msg
3112 000099E8 58                        <1>       pop    eax
3113                                    <1>       ;pop   ax
3114 000099E9 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 000099EA 803E20                    <1>       cmp    byte [esi], 20h
3128 000099ED 7701                      <1>         ja   short loc_rmdir_parse_path_name
3129                                    <1>
3130                                    <1> loc_rmdir_nodirname_retn:
3131 000099EF C3                        <1>       retn
3132                                    <1>
3133                                    <1> loc_rmdir_parse_path_name:
3134 000099F0 BF[F68A0100]              <1>       mov    edi, FindFile_Drv
3135 000099F5 E8481C0000                <1>       call   parse_path_name
3136 000099FA 0F8261F5FFFF              <1>       jc     loc_cmd_failed
3137                                    <1>
3138                                    <1> loc_rmdir_check_dirname_exists:
3139 00009A00 BE[388B0100]              <1>       mov    esi, FindFile_Name
3140 00009A05 803E20                    <1>       cmp    byte [esi], 20h
3141 00009A08 0F8653F5FFFF              <1>       jna    loc_cmd_failed
3142 00009A0E 8935[B48B0100]            <1>       mov    [DelFile_FNPointer], esi
3143                                    <1>
3144                                    <1> loc_rmdir_drv:
3145 00009A14 8A35[56820100]            <1>       mov    dh, [Current_Drv]
3146 00009A1A 8835[B2890100]            <1>       mov    [RUN_CDRV], dh
3147                                    <1>
3148 00009A20 8A15[F68A0100]            <1>       mov    dl, [FindFile_Drv]
3149 00009A26 38F2                      <1>       cmp    dl, dh
3150 00009A28 740B                      <1>       je     short loc_rmdir_change_directory
3151                                    <1>
3152 00009A2A E89FE6FFFF                <1>       call   change_current_drive
3153 00009A2F 0F826AFEFFFF              <1>       jc     loc_file_rw_cmd_failed
3154                                    <1>
3155                                    <1> loc_rmdir_change_directory:
3156 00009A35 803D[F78A0100]20          <1>       cmp    byte [FindFile_Directory], 20h
3157 00009A3C 7614                      <1>       jna    short loc_rmdir_find_directory
3158                                    <1>
3159 00009A3E FE05[23380100]            <1>       inc    byte [Restore_CDIR]
3160 00009A44 BE[F78A0100]              <1>       mov    esi, FindFile_Directory
3161 00009A49 30E4                      <1>       xor    ah, ah ; CD_COMMAND sign -> 0
3162 00009A4B E8DC150000                <1>       call   change_current_directory
3163 00009A50 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 00009A52 8B35[B48B0100]            <1>       mov    esi, [DelFile_FNPointer]
3171 00009A58 66B81008                  <1>       mov    ax, 0810h ; Only directories
3172 00009A5C E810FAFFFF                <1>       call   find_first_file
3173 00009A61 730A                      <1>       jnc    short loc_rmdir_ambgfn_check
3174                                    <1>
3175                                    <1> loc_rmdir_check_error_code:
3176 00009A63 3C02                      <1>       cmp    al, 2
3177 00009A65 740B                      <1>       je     short loc_rmdir_directory_not_found
3178 00009A67 F9                        <1>       stc
3179 00009A68 E932FEFFFF                <1>       jmp    loc_file_rw_cmd_failed
3180                                    <1>
3181                                    <1> loc_rmdir_ambgfn_check:
3182 00009A6D 6621D2                    <1>       and    dx, dx ; Ambiguous filename chars used sign (DX>0)
3183 00009A70 740F                      <1>       jz     short loc_rmdir_directory_found
3184                                    <1>
3185                                    <1> loc_rmdir_directory_not_found:
3186 00009A72 BE[253B0100]              <1>       mov    esi, Msg_Dir_Not_Found
3187 00009A77 E8C9DAFFFF                <1>       call   print_msg
3188                                    <1>
3189 00009A7C E91EFEFFFF                <1>       jmp    loc_file_rw_restore_retn
3190                                    <1>
3191                                    <1> loc_rmdir_directory_found:
3192 00009A81 80E307                    <1>       and    bl, 07h ; Attributes
3193 00009A84 0F851FFEFFFF              <1>       jnz    loc_permission_denied
3194                                    <1>
3195                                    <1> loc_rmdir_save_lnel: ; 28/02/2016
3196                                    <1>        ;mov  bh, [LongName_EntryLength]
3197 00009A8A 883D[BE8B0100]            <1>       mov    [DelFile_LNEL], bh ; Long name entry length (if > 0)
3198                                    <1>       ; edi = Directory Entry Offset (DirBuff)
3199                                    <1>       ; esi = Directory Entry (FFF Structure)
3200                                    <1>       ;mov   [DelFile_DirEntryAddr], edi ; not required
3201                                    <1>       ;mov   ax, [edi+20] ; First Cluster High Word
3202                                    <1>         ;shl eax, 16
3203                                    <1>       ;mov   ax, [edi+26] ; First Cluster Low Word
3204                                    <1>       ; ROOT Dir First Cluster = 0
3205                                    <1>         ;cmp eax, 2
3206                                    <1>       ;jb    loc_update_direntry_1
3207                                    <1>
3208                                    <1> pass_rmdir_fc_check:
3209 00009A90 57                        <1>       push   edi ; * (29/02/2016)
3210                                    <1>
3211 00009A91 BE[BB3C0100]              <1>       mov    esi, Msg_DoYouWantRmDir
3212 00009A96 E8AADAFFFF                <1>       call   print_msg
```

```
3213 00009A9B 8B35[B48B0100]      <1>       mov    esi, [DelFile_FNPointer]
3214 00009AA1 E89FDAFFFF          <1>       call   print_msg
3215 00009AA6 BE[A73C0100]        <1>       mov    esi, Msg_YesNo
3216 00009AAB E895DAFFFF          <1>       call   print_msg
3217                              <1>
3218                              <1> loc_rmdir_ask_again:
3219 00009AB0 30E4                <1>       xor    ah, ah
3220 00009AB2 E82E74FFFF          <1>       call   int16h
3221 00009AB7 3C1B                <1>       cmp    al, 1Bh
3222                              <1>       ;je    short loc_do_not_delete_directory
3223 00009AB9 7433                <1>         je      loc_rmdir_y_n_escape ; 06/03/2016
3224 00009ABB 24DF                <1>       and    al, 0DFh
3225 00009ABD A2[B13C0100]        <1>       mov    [Y_N_nextline], al
3226 00009AC2 3C59                <1>       cmp    al, 'Y'
3227 00009AC4 7404                <1>       je     short loc_rmdir_yes_delete_directory
3228 00009AC6 3C4E                <1>       cmp    al, 'N'
3229 00009AC8 75E6                <1>       jne    short loc_rmdir_ask_again
3230                              <1>
3231                              <1> loc_do_not_delete_directory:
3232                              <1> loc_rmdir_yes_delete_directory:
3233 00009ACA E809FFFFFF          <1>       call   y_n_answer ; 29/12/2017
3234 00009ACF 5F                  <1>       pop    edi ; * (29/02/2016)
3235                              <1>       ;cmp   al, 'Y' ; 'yes'
3236                              <1>       ;cmc
3237                              <1>         ;jnc loc_file_rw_restore_retn
3238 00009AD0 3C4E                <1>       cmp    al, 'N' ; 'no'
3239 00009AD2 0F84C7FDFFFF        <1>         je   loc_file_rw_restore_retn
3240                              <1>
3241                              <1>       ; 29/12/2017
3242 00009AD8 E869000000          <1>       call   delete_sub_directory
3243 00009ADD 7213                <1>       jc     short loc_rmdir_cmd_failed
3244                              <1>
3245                              <1> loc_rmdir_ok:
3246 00009ADF BE[B53C0100]        <1>       mov    esi, Msg_OK
3247 00009AE4 E85CDAFFFF          <1>       call   print_msg
3248 00009AE9 E9B1FDFFFF          <1>       jmp    loc_file_rw_restore_retn
3249                              <1>
3250                              <1> loc_rmdir_y_n_escape:
3251 00009AEE B04E                <1>       mov    al, 'N' ; 'no'
3252 00009AF0 EBD8                <1>         jmp     loc_do_not_delete_directory
3253                              <1>
3254                              <1> loc_rmdir_cmd_failed:
3255                              <1>       ; 29/12/2017
3256 00009AF2 09C0                <1>       or     eax, eax ; EAX = 0 -> Directory not empty!
3257 00009AF4 7426                <1>       jz     short loc_rmdir_directory_not_empty
3258                              <1>
3259                              <1>       ; EAX > 0 -> Error code in AL (or AX or EAX)
3260                              <1>
3261 00009AF6 833D[72890100]01    <1>       cmp    dword [FAT_ClusterCounter], 1
3262 00009AFD 0F829CFDFFFF        <1>       jb     loc_file_rw_cmd_failed
3263 00009B03 F9                  <1>       stc
3264                              <1> loc_rmdir_cmd_return:
3265                              <1>       ; 01/03/2016
3266 00009B04 9C                  <1>       pushf
3267                              <1>       ; ESI = Logical DOS Drive Description Table address
3268 00009B05 66BB00FF            <1>       mov    bx, 0FF00h ; BH = FFh -> use ESI for Drive parameters
3269                              <1>                   ; BL = 0 -> Recalculate free cluster count
3270 00009B09 50                  <1>       push   eax
3271 00009B0A E8C3380000          <1>       call   calculate_fat_freespace
3272 00009B0F 58                  <1>       pop    eax
3273 00009B10 9D                  <1>       popf
3274 00009B11 0F8288FDFFFF        <1>       jc     loc_file_rw_cmd_failed
3275 00009B17 E983FDFFFF          <1>       jmp    loc_file_rw_restore_retn
3276                              <1>
3277                              <1> loc_rmdir_directory_not_empty:
3278 00009B1C BE[DC3C0100]        <1>       mov    esi, Msg_Dir_Not_Empty
3279 00009B21 E81FDAFFFF          <1>       call   print_msg
3280                              <1>       ; 01/03/2016
3281 00009B26 A1[72890100]        <1>       mov    eax, [FAT_ClusterCounter]
3282 00009B2B 09C0                <1>       or     eax, eax ; 0 ?
3283 00009B2D 0F846CFDFFFF        <1>       jz     loc_file_rw_restore_retn
3284                              <1>       ; ESI = Logical DOS Drive Description Table address
3285 00009B33 66BB01FF            <1>       mov    bx, 0FF01h ; BH = FFh -> use ESI for Drive parameters
3286                              <1>                   ; BL = 1 -> add free clusters
3287 00009B37 E896380000          <1>       call   calculate_fat_freespace
3288 00009B3C 09C9                <1>       or     ecx, ecx
3289 00009B3E 0F845BFDFFFF        <1>         jz      loc_file_rw_restore_retn ; ecx = 0 -> OK
3290                              <1>       ; ecx > 0 -> Error (Recalculation is needed)
3291 00009B44 EBBE                <1>       jmp    short loc_rmdir_cmd_return
3292                              <1>
3293                              <1>
3294                              <1> delete_sub_directory:
3295                              <1>       ; 29/12/2017
3296                              <1>       ; (moved here from 'delete_directory' for 'sysrmdir' )
3297                              <1>
3298                              <1>       ; EDI = Directory buffer entry offset/address
3299                              <1>
3300                              <1> loc_rmdir_delete_short_name_check_dir_empty:
3301 00009B46 668B4714            <1>       mov    ax, [edi+20] ; First Cluster High Word
3302 00009B4A C1E010              <1>         shl  eax, 16
3303 00009B4D 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 00009B51 29DB                <1>       sub    ebx, ebx
3312                              <1>       ; 29/12/2017
3313 00009B53 891D[72890100]      <1>       mov    [FAT_ClusterCounter], ebx ; 0 ; Reset
3314                              <1>
3315 00009B59 8A3D[F68A0100]      <1>       mov    bh, [FindFile_Drv]
3316 00009B5F BE00010900          <1>       mov    esi, Logical_DOSDisks
3317 00009B64 01DE                <1>       add    esi, ebx
```

```
3318                                    <1>
3319 00009B66 66817F0CA101              <1>       cmp    word [edi+DirEntry_NTRes], 01A1h
3320 00009B6C 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 00009B6E 83F802                    <1>       cmp    eax, 2
3327 00009B71 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 00009B73 B813000000                <1>       mov    eax, ERR_NOT_DIR ; not a valid directory!
3332                                    <1>       ;stc
3333 00009B78 C3                        <1>       retn
3334                                    <1>
3335                                    <1> loc_rmdir_get_last_cluster_1:
3336 00009B79 807E0303                  <1>       cmp    byte [esi+LD_FATType], 3 ; FAT32
3337 00009B7D 750C                      <1>       jne    short loc_rmdir_get_last_cluster_2
3338                                    <1>
3339                                    <1>       ; is it root directory ?
3340 00009B7F 3B4632                    <1>       cmp    eax, [esi+LD_BPB+BPB_RootClus]
3341 00009B82 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 00009B84 B80B000000                <1>       mov    eax, ERR_PERM_DENIED ; permission denied!
3346 00009B89 F9                        <1>       stc
3347 00009B8A C3                        <1>       retn
3348                                    <1>
3349                                    <1> loc_rmdir_get_last_cluster_2:
3350                                    <1>       ; 29/12/2017
3351 00009B8B A3[B88B0100]              <1>       mov    [DelFile_FCluster], eax
3352                                    <1>
3353                                    <1>       ;mov   dx, [DirBuff_EntryCounter]
3354 00009B90 668B15[708B0100]          <1>       mov    dx, [FindFile_DirEntryNumber] ; 27/02/2016
3355 00009B97 668915[BC8B0100]          <1>       mov    [DelFile_EntryCounter], dx
3356                                    <1>
3357 00009B9E 8B15[81890100]            <1>       mov    edx, [DirBuff_Cluster]
3358 00009BA4 8915[E88B0100]            <1>       mov    [RmDir_ParentDirCluster], edx
3359                                    <1>
3360 00009BAA 893D[E48B0100]            <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 00009BB0 E89C390000                <1>       call   get_last_cluster
3367                                    <1>         ;jc  loc_rmdir_cmd_failed
3368 00009BB5 721E                      <1>       jc     short loc_delete_sub_dir_retn ; 29/12/2017
3369                                    <1>
3370 00009BB7 3B05[B88B0100]            <1>       cmp    eax, [DelFile_FCluster]
3371 00009BBD 7517                      <1>       jne    short loc_rmdir_multi_dir_clusters
3372                                    <1>
3373 00009BBF C605[E38B0100]00          <1>       mov    byte [RmDir_MultiClusters], 0
3374 00009BC6 EB15                      <1>       jmp    short pass_rmdir_multi_dir_clusters
3375                                    <1>
3376                                    <1> loc_rmdir_check_fs_directory:
3377                                    <1>       ; 29/12/2017
3378 00009BC8 807E04A1                  <1>       cmp    byte [esi+LD_FSType], 0A1h
3379 00009BCC 75B6                      <1>       jne    short loc_rmdir_permission_denied
3380                                    <1>
3381                                    <1> loc_rmdir_delete_fs_directory:
3382 00009BCE E876130000                <1>       call   delete_fs_directory
3383                                    <1>       ;jnc   loc_print_deleted_message
3384 00009BD3 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 00009BD5 C3                        <1>       retn
3396                                    <1>
3397                                    <1> loc_rmdir_multi_dir_clusters:
3398 00009BD6 C605[E38B0100]01          <1>       mov    byte [RmDir_MultiClusters], 1
3399                                    <1>
3400                                    <1> pass_rmdir_multi_dir_clusters:
3401 00009BDD A3[EC8B0100]              <1>       mov    [RmDir_DirLastCluster], eax
3402 00009BE2 890D[F08B0100]            <1>       mov    [RmDir_PreviousCluster], ecx
3403                                    <1>
3404                                    <1> loc_rmdir_load_fat_sub_directory:
3405 00009BE8 E84F330000                <1>       call   load_FAT_sub_directory
3406                                    <1>       ;jc    loc_rmdir_cmd_failed
3407 00009BED 72E6                      <1>       jc     short loc_delete_sub_dir_retn
3408                                    <1>
3409                                    <1> loc_rmdir_find_last_dir_entry:
3410 00009BEF 56                        <1>       push   esi
3411 00009BF0 BE[DA8A0100]              <1>       mov    esi, Dir_File_Name
3412 00009BF5 C6062A                    <1>       mov    byte [esi], '*'
3413 00009BF8 C646082A                  <1>       mov    byte [esi+8], '*'
3414 00009BFC 31DB                      <1>       xor    ebx, ebx ; Entry offset  = 0
3415                                    <1> loc_rmdir_find_last_dir_entry_next:
3416 00009BFE 66B80008                  <1>       mov    ax, 0800h ; Except volume/long names
3417 00009C02 6631C9                    <1>       xor    cx, cx ; 0 = Find a valid file or dir name
3418 00009C05 E87A170000                <1>       call   find_directory_entry
3419 00009C0A 7225                      <1>       jc     short loc_rmdir_empty_dir_cluster
3420 00009C0C 83FB01                    <1>       cmp    ebx, 1
3421 00009C0F 771B                      <1>       ja     short loc_rmdir_directory_not_empty_1
3422                                    <1> loc_rmdir_dot_entry_check:
```

```
3423 00009C11 80FD2E              <1>       cmp    ch, '.' ; The first char of the dir entry
3424 00009C14 7516                <1>       jne    short loc_rmdir_directory_not_empty_1
3425 00009C16 08DB                <1>       or     bl, bl
3426 00009C18 7506                <1>       jnz    short loc_rmdir_dotdot_entry_check
3427 00009C1A 807F0120            <1>       cmp    byte [edi+1], 20h
3428 00009C1E EB06                <1>       jmp    short pass_rmdir_dot_entry_check
3429                              <1>
3430                              <1> loc_rmdir_dotdot_entry_check:
3431 00009C20 66817F012E20        <1>       cmp    word [edi+1], '. '
3432                              <1> pass_rmdir_dot_entry_check:
3433 00009C26 7504                <1>       jne    short loc_rmdir_directory_not_empty_1
3434 00009C28 FEC3                <1>       inc    bl
3435 00009C2A EBD2                <1>       jmp    short loc_rmdir_find_last_dir_entry_next
3436                              <1>
3437                              <1> loc_rmdir_directory_not_empty_1:
3438 00009C2C 58                  <1>       pop    eax ; pushed esi
3439 00009C2D 31C0                <1>       xor    eax, eax ; 0
3440                              <1> loc_rmdir_directory_not_empty_2:
3441                              <1> loc_delete_sub_dir_stc_retn:
3442 00009C2F F9                  <1>       stc
3443 00009C30 C3                  <1>       retn
3444                              <1>
3445                              <1> loc_rmdir_empty_dir_cluster:
3446 00009C31 5E                  <1>       pop    esi
3447                              <1>
3448                              <1> loc_rmdir_set_prev_cluster_dir_last_cluster:
3449 00009C32 803D[E38B0100]00    <1>       cmp    byte [RmDir_MultiClusters], 0
3450 00009C39 7613                <1>       jna    short loc_rmdir_unlink_dir_last_cluster
3451                              <1>
3452 00009C3B A1[F08B0100]        <1>       mov    eax, [RmDir_PreviousCluster]
3453                              <1>       ;xor   ecx, ecx
3454 00009C40 49                  <1>       dec    ecx ; FFFFFFFFh
3455 00009C41 E83A340000          <1>       call   update_cluster
3456 00009C46 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 00009C48 21C0                <1>       and    eax, eax
3465 00009C4A 75E3                <1>       jnz    short loc_delete_sub_dir_stc_retn
3466 00009C4C EB12                <1>       jmp    short loc_rmdir_save_fat_buffer
3467                              <1>
3468                              <1> loc_rmdir_unlink_dir_last_cluster:
3469 00009C4E A1[EC8B0100]        <1>       mov    eax, [RmDir_DirLastCluster]
3470 00009C53 31C9                <1>       xor    ecx, ecx ; 0
3471 00009C55 E826340000          <1>       call   update_cluster
3472 00009C5A 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 00009C5C 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 00009C5E 75CF                <1>       jnz    short loc_delete_sub_dir_stc_retn
3481                              <1>
3482                              <1> loc_rmdir_save_fat_buffer:
3483 00009C60 803D[6A890100]02    <1>       cmp    byte [FAT_BuffValidData], 2
3484 00009C67 7528                <1>       jne    short loc_rmdir_calculate_FAT_freespace
3485 00009C69 E8CF360000          <1>       call   save_fat_buffer
3486                              <1>       ;jc    short loc_rmdir_cmd_failed
3487                              <1>       ; 29/12/2017
3488 00009C6E 7219                <1>       jc     short loc_rmdir_unlink_error_retn
3489                              <1>
3490                              <1>       ; 01/03/2016
3491 00009C70 803D[E38B0100]00    <1>       cmp    byte [RmDir_MultiClusters], 0
3492 00009C77 7618                <1>       jna    short loc_rmdir_calculate_FAT_freespace
3493                              <1>
3494 00009C79 A1[B88B0100]        <1>       mov    eax, [DelFile_FCluster]
3495 00009C7E 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 00009C83 B81C000000          <1>       mov    eax, ERR_INV_FORMAT ; 28 = Invalid format
3500                              <1> loc_rmdir_unlink_stc_retn:
3501 00009C88 F9                  <1>       stc
3502                              <1> loc_rmdir_unlink_error_retn:
3503 00009C89 C3                  <1>       retn
3504                              <1>
3505                              <1> loc_rmdir_delete_short_name_invalid_data:
3506 00009C8A 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 00009C8F EBF7                <1>       jmp    short loc_rmdir_unlink_stc_retn
3511                              <1>
3512                              <1> loc_rmdir_calculate_FAT_freespace:
3513                              <1>       ;mov   eax, [FAT_ClusterCounter]
3514                              <1>       ; 29/12/2017
3515 00009C91 29C0                <1>       sub    eax, eax ; 0
3516 00009C93 8705[72890100]      <1>       xchg   eax, [FAT_ClusterCounter]
3517                              <1>       ;
3518 00009C99 66BB01FF            <1>       mov    bx, 0FF01h
3519                              <1>       ; BL = 1 -> Add EAX to free space count
3520                              <1>       ; BH = FFh ->
3521                              <1>       ; ESI = Logical DOS Drive Description Table address
3522 00009C9D E830370000          <1>       call   calculate_fat_freespace
3523                              <1>
3524 00009CA2 21C9                <1>       and    ecx, ecx ; ecx = 0 -> valid free sector count
3525 00009CA4 7409                <1>       jz     short loc_rmdir_delete_short_name_continue
3526                              <1>
3527                              <1> loc_rmdir_recalculate_FAT_freespace:
```

```
3528 00009CA6 66BB00FF            <1>         mov     bx, 0FF00h ; BL = 0 -> Recalculate free space
3529 00009CAA E823370000          <1>       call   calculate_fat_freespace
3530                              <1>
3531                              <1> loc_rmdir_delete_short_name_continue:
3532 00009CAF A1[E88B0100]        <1>       mov    eax, [RmDir_ParentDirCluster]
3533 00009CB4 83F802              <1>       cmp    eax, 2
3534 00009CB7 7309                <1>       jnb    short loc_rmdir_del_short_name_load_sub_dir
3535 00009CB9 E8F3310000          <1>       call   load_FAT_root_directory
3536                              <1>       ;jc    loc_file_rw_cmd_failed
3537                              <1>       ; 29/12/2017
3538 00009CBE 72C9                <1>       jc     short loc_rmdir_unlink_error_retn
3539 00009CC0 EB07                <1>       jmp    short loc_rmdir_del_short_name_ld_chk_fclust
3540                              <1>
3541                              <1> loc_rmdir_del_short_name_load_sub_dir:
3542 00009CC2 E875320000          <1>       call   load_FAT_sub_directory
3543                              <1>       ;jc    loc_file_rw_cmd_failed
3544                              <1>       ; 29/12/2017
3545 00009CC7 72C0                <1>       jc     short loc_rmdir_unlink_error_retn
3546                              <1>
3547                              <1> loc_rmdir_del_short_name_ld_chk_fclust:
3548 00009CC9 0FB73D[E48B0100]    <1>       movzx  edi, word [RmDir_DirEntryOffset]
3549 00009CD0 81C700000800        <1>       add    edi, Directory_Buffer
3550                              <1>
3551 00009CD6 668B4714            <1>       mov    ax, [edi+20] ; First Cluster High Word
3552 00009CDA C1E010              <1>       shl    eax, 16
3553 00009CDD 668B471A            <1>       mov    ax, [edi+26] ; First Cluster Low Word
3554                              <1>         ; Not necessary...
3555 00009CE1 3B05[B88B0100]      <1>       cmp    eax, [DelFile_FCluster]
3556 00009CE7 75A1                <1>       jne    short loc_rmdir_delete_short_name_invalid_data
3557                              <1>       ;
3558 00009CE9 C607E5              <1>       mov    byte [edi], 0E5h ; 'Deleted' sign
3559                              <1>       ; 27/02/2016
3560                              <1>       ; TRDOS v1 has a bug here! it does not set
3561                              <1>       ; 'DirBuff_ValidData' to 2; as result of this bug,
3562                              <1>       ; 'save_directory_buffer' would not save the change !
3563 00009CEC C605[7C890100]02    <1>       mov    byte [DirBuff_ValidData], 2 ; change sign
3564                              <1>       ;
3565 00009CF3 E8AE1D0000          <1>       call   save_directory_buffer
3566                              <1>       ;jc    loc_file_rw_cmd_failed
3567                              <1>       ; 29/12/2017
3568 00009CF8 728F                <1>       jc     short loc_rmdir_unlink_error_retn
3569                              <1>
3570                              <1> loc_rmdir_del_long_name:
3571 00009CFA 0FB615[BE8B0100]    <1>       movzx  edx, byte [DelFile_LNEL]
3572 00009D01 08D2                <1>       or     dl, dl
3573 00009D03 7410                <1>       jz     short loc_rmdir_update_parent_dir_lmdt
3574                              <1>
3575 00009D05 0FB705[BC8B0100]    <1>       movzx  eax, word [DelFile_EntryCounter]
3576 00009D0C 29D0                <1>       sub    eax, edx
3577                              <1>       ; 29/12/2017
3578 00009D0E 7205                <1>       jc     short loc_rmdir_update_parent_dir_lmdt
3579                              <1>
3580                              <1>       ; EAX = Directory Entry Number of the long name last entry
3581 00009D10 E8EF1E0000          <1>       call   delete_longname
3582                              <1>
3583                              <1> loc_rmdir_update_parent_dir_lmdt:
3584 00009D15 E8271E0000          <1>       call   update_parent_dir_lmdt
3585                              <1>       ;jc    short loc_file_rw_cmd_failed
3586                              <1>       ; 29/12/2017
3587                              <1>       ;jc    short loc_rmdir_unlink_error_retn
3588                              <1>
3589                              <1> loc_delete_sub_directory_ok:
3590                              <1>       ; 29/12/2017
3591 00009D1A 31C0                <1>       xor    eax, eax ;  0 ;  cf = 0
3592 00009D1C C3                  <1>       retn
3593                              <1>
3594                              <1>
3595                              <1> delete_file:
3596                              <1>       ; 29/02/2016
3597                              <1>       ; 28/02/2016 (TRDOS 386 = TRDOS v2.0)
3598                              <1>       ; 09/08/2010 (CMD_INTR.ASM, 'cmp_cmd_del')
3599                              <1>       ; 28/02/2010
3600                              <1>
3601                              <1> get_delfile_fchar:
3602                              <1>       ; esi = file name
3603 00009D1D 803E20              <1>       cmp    byte [esi], 20h
3604 00009D20 7701                <1>         ja   short loc_delfile_parse_path_name
3605                              <1>
3606                              <1> loc_delfile_nofilename_retn:
3607 00009D22 C3                  <1>       retn
3608                              <1>
3609                              <1> loc_delfile_parse_path_name:
3610 00009D23 BF[F68A0100]        <1>       mov    edi, FindFile_Drv
3611 00009D28 E815190000          <1>       call   parse_path_name
3612 00009D2D 0F822EF2FFFF        <1>       jc     loc_cmd_failed
3613                              <1>
3614                              <1> loc_delfile_check_filename_exists:
3615 00009D33 BE[388B0100]        <1>       mov    esi, FindFile_Name
3616 00009D38 803E20              <1>       cmp    byte [esi], 20h
3617 00009D3B 0F8620F2FFFF        <1>       jna    loc_cmd_failed
3618 00009D41 8935[B48B0100]      <1>       mov    [DelFile_FNPointer], esi
3619                              <1>
3620                              <1> loc_delfile_drv:
3621 00009D47 8A15[F68A0100]      <1>       mov    dl, [FindFile_Drv]
3622 00009D4D 8A35[56820100]      <1>       mov    dh, [Current_Drv]
3623 00009D53 8835[B2890100]      <1>       mov    [RUN_CDRV], dh
3624 00009D59 38F2                <1>       cmp    dl, dh
3625 00009D5B 740B                <1>       je     short loc_delfile_change_directory
3626                              <1>
3627 00009D5D E86CE3FFFF          <1>       call   change_current_drive
3628 00009D62 0F8237FBFFFF        <1>       jc     loc_file_rw_cmd_failed
3629                              <1>
3630                              <1> loc_delfile_change_directory:
3631 00009D68 803D[F78A0100]20    <1>       cmp    byte [FindFile_Directory], 20h
3632 00009D6F 7618                <1>       jna    short loc_delfile_find
```

```
                                <1>
00009D71 FE05[23380100]         <1>      inc    byte [Restore_CDIR]
00009D77 BE[F78A0100]           <1>      mov    esi, FindFile_Directory
00009D7C 30E4                   <1>      xor    ah, ah ; CD_COMMAND sign -> 0
00009D7E E8A9120000             <1>      call   change_current_directory
00009D83 0F8216FBFFFF           <1>      jc     loc_file_rw_cmd_failed
                                <1>
                                <1> ;loc_delfile_change_prompt_dir_string:
                                <1>      ;call  change_prompt_dir_string
                                <1>
                                <1> loc_delfile_find:
                                <1>      ;mov   esi, FindFile_Name
00009D89 8B35[B48B0100]         <1>      mov    esi, [DelFile_FNPointer]
00009D8F 66B80018               <1>      mov    ax, 1800h ; Except volume label and dirs
00009D93 E8D9F6FFFF             <1>      call   find_first_file
00009D98 0F8201FBFFFF           <1>      jc     loc_file_rw_cmd_failed
                                <1>
                                <1> loc_delfile_ambgfn_check:
00009D9E 6621D2                 <1>      and    dx, dx ; Ambiguous filename chars used sign (DX>0)
00009DA1 740B                   <1>      jz     short loc_delfile_found
                                <1>
                                <1> loc_file_not_found:
00009DA3 B802000000             <1>      mov    eax, 2 ; File not found sign
00009DA8 F9                     <1>      stc
00009DA9 E9F1FAFFFF             <1>      jmp    loc_file_rw_cmd_failed
                                <1>
                                <1> loc_delfile_found:
00009DAE 80E307                 <1>      and    bl, 07h ; Attributes
00009DB1 0F85F2FAFFFF           <1>        jnz     loc_permission_denied
                                <1>
                                <1> ;loc_delfile_found_save_lnel:
                                <1> ;     mov    [DelFile_LNEL], bh ; Long name entry length (if > 0)
                                <1>
                                <1> loc_delfile_ask_for_delete:
00009DB7 57                     <1>      push   edi ; * (29/02/2016)
                                <1>
00009DB8 BE[F33C0100]           <1>      mov    esi, Msg_DoYouWantDelete
00009DBD E883D7FFFF             <1>      call   print_msg
00009DC2 8B35[B48B0100]         <1>      mov    esi, [DelFile_FNPointer]
00009DC8 E878D7FFFF             <1>      call   print_msg
00009DCD BE[A73C0100]           <1>      mov    esi, Msg_YesNo
00009DD2 E86ED7FFFF             <1>      call   print_msg
                                <1>
                                <1> loc_delfile_ask_again:
00009DD7 30E4                   <1>      xor    ah, ah
00009DD9 E80771FFFF             <1>      call   int16h
00009DDE 3C1B                   <1>      cmp    al, 1Bh
                                <1>      ;je    short loc_do_not_delete_file
00009DE0 7449                   <1>      je     short loc_delfile_y_n_escape ; 06/03/2016
00009DE2 24DF                   <1>      and    al, 0DFh
00009DE4 A2[B13C0100]           <1>      mov    [Y_N_nextline], al
00009DE9 3C59                   <1>      cmp    al, 'Y'
00009DEB 7404                   <1>      je     short loc_yes_delete_file
00009DED 3C4E                   <1>      cmp    al, 'N'
00009DEF 75E6                   <1>      jne    short loc_delfile_ask_again
                                <1>
                                <1> loc_do_not_delete_file:
                                <1> loc_yes_delete_file:
00009DF1 E8E2FBFFFF             <1>      call   y_n_answer ; 29/12/2017
00009DF6 5F                     <1>      pop    edi ; * (29/02/2016)
                                <1>      ;cmp   al, 'Y' ; 'yes'
                                <1>      ;cmc
                                <1>        ;jnc loc_file_rw_restore_retn
00009DF7 3C4E                   <1>      cmp    al, 'N' ; 'no'
00009DF9 0F84A0FAFFFF           <1>        je   loc_file_rw_restore_retn
                                <1>
                                <1> loc_delete_file:
00009DFF 8A3D[F68A0100]         <1>      mov    bh, [FindFile_Drv]
                                <1>      ;mov   bl, [DelFile_LNEL]
00009E05 8A1D[458B0100]         <1>      mov    bl, [FindFile_LongNameEntryLength]
                                <1>      ;mov   cx, [DirBuff_EntryCounter]
00009E0B 668B0D[708B0100]       <1>      mov    cx, [FindFile_DirEntryNumber]
                                <1>      ; (*) EDI = Directory buffer entry offset/address
00009E12 E8D71F0000             <1>      call   remove_file ; (FILE.ASM, 'proc_delete_file')
00009E17 0F8378FAFFFF           <1>      jnc    loc_print_deleted_message
                                <1>
                                <1>      ;cmp   al, 05h
00009E1D 3C0B                   <1>      cmp    al, ERR_PERM_DENIED  ; 29/12/2017 (5 -> 11)
00009E1F 0F8484FAFFFF           <1>      je     loc_permission_denied
00009E25 F9                     <1>      stc
00009E26 E974FAFFFF             <1>      jmp    loc_file_rw_cmd_failed
                                <1>
                                <1> loc_delfile_y_n_escape:
00009E2B B04E                   <1>      mov    al, 'N' ; 'no'
00009E2D EBC2                   <1>      jmp    short loc_do_not_delete_file
                                <1>
                                <1> set_file_attributes:
                                <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
00009E2F 6631C0                 <1>      xor    ax, ax
00009E32 66A3[443D0100]         <1>      mov    [Attr_Chars], ax
00009E38 A2[0C8C0100]           <1>      mov    [Attributes], al
                                <1>
                                <1> get_attrib_fchar:
                                <1>      ; esi = file name
00009E3D 8A06                   <1>      mov    al, [esi]
00009E3F 3C20                   <1>      cmp    al, 20h
00009E41 7623                   <1>      jna    short loc_attr_file_nofilename_retn
                                <1>
                                <1> loc_scan_attrib_params:
```

```
3738 00009E43 3C2D              <1>       cmp    al, '-'
3739 00009E45 0F871C010000      <1>       ja     loc_attr_file_parse_path_name
3740 00009E4B 7408              <1>       je     short loc_attr_space
3741                            <1>
3742 00009E4D 3C2B              <1>       cmp    al, '+'
3743 00009E4F 0F850CF1FFFF      <1>       jne    loc_cmd_failed
3744                            <1>
3745                            <1> loc_attr_space:
3746 00009E55 8A6601            <1>       mov    ah, [esi+1]
3747 00009E58 80FC20            <1>       cmp    ah, 20h
3748 00009E5B 770A              <1>       ja     short pass_attr_space
3749 00009E5D 0F82FEF0FFFF      <1>       jb     loc_cmd_failed
3750 00009E63 46                <1>       inc    esi
3751 00009E64 EBEF              <1>       jmp    short loc_attr_space
3752                            <1>
3753                            <1> loc_attr_file_nofilename_retn:
3754 00009E66 C3                <1>       retn
3755                            <1>
3756                            <1> pass_attr_space:
3757 00009E67 80E4DF            <1>       and    ah, 0DFh
3758 00009E6A 80FC53            <1>       cmp    ah, 'S'
3759 00009E6D 0F87EEF0FFFF      <1>       ja     loc_cmd_failed
3760 00009E73 7204              <1>       jb     short pass_attr_system
3761 00009E75 B404              <1>       mov    ah, 04h   ; System
3762 00009E77 EB21              <1>       jmp    short pass_attr_archive
3763                            <1>
3764                            <1> pass_attr_system:
3765 00009E79 80FC48            <1>       cmp    ah, 'H'
3766 00009E7C 7706              <1>       ja     short pass_attr_hidden
3767 00009E7E 7213              <1>       jb     short pass_attr_read_only
3768 00009E80 B402              <1>       mov    ah, 02h    ; Hidden
3769 00009E82 EB16              <1>       jmp    short pass_attr_archive
3770                            <1>
3771                            <1> pass_attr_hidden:
3772 00009E84 80FC52            <1>       cmp    ah, 'R'
3773 00009E87 0F87D4F0FFFF      <1>       ja     loc_cmd_failed
3774 00009E8D 7204              <1>       jb     short pass_attr_read_only ; Read only
3775 00009E8F B401              <1>       mov    ah, 01h
3776 00009E91 EB07              <1>       jmp    short pass_attr_archive
3777                            <1>
3778                            <1> pass_attr_read_only:
3779 00009E93 80FC41            <1>       cmp    ah, 'A'
3780 00009E96 753B              <1>       jne    short loc_chk_attr_enter
3781 00009E98 B420              <1>       mov    ah, 20h    ; Archive
3782                            <1>
3783                            <1> pass_attr_archive:
3784 00009E9A 3C2D              <1>       cmp    al, '-'
3785 00009E9C 7508              <1>       jne    short pass_reducing_attributes
3786 00009E9E 0825[443D0100]    <1>       or     [Attr_Chars], ah
3787 00009EA4 EB06              <1>       jmp    short loc_change_attributes_inc
3788                            <1>
3789                            <1> pass_reducing_attributes:
3790 00009EA6 0825[453D0100]    <1>       or     [Attr_Chars+1], ah
3791                            <1>
3792                            <1> loc_change_attributes_inc:
3793 00009EAC 46                <1>       inc    esi
3794 00009EAD 8A6601            <1>       mov    ah, [esi+1]
3795 00009EB0 80FC20            <1>       cmp    ah, 20h
3796 00009EB3 7227              <1>       jb     short pass_change_attr
3797 00009EB5 74F5              <1>       je     short loc_change_attributes_inc
3798 00009EB7 80FC2D            <1>       cmp    ah, '-'
3799 00009EBA 770D              <1>       ja     short loc_chk_next_attr_char1
3800 00009EBC 7405              <1>       je     short loc_chk_next_attr_char0
3801 00009EBE 80FC2B            <1>       cmp    ah, '+'
3802 00009EC1 7506              <1>       jne    short loc_chk_next_attr_char1
3803                            <1>
3804                            <1> loc_chk_next_attr_char0:
3805 00009EC3 46                <1>       inc    esi
3806 00009EC4 668B06            <1>       mov    ax, [esi]
3807 00009EC7 EB9E              <1>       jmp    short pass_attr_space
3808                            <1>
3809                            <1> loc_chk_next_attr_char1:
3810 00009EC9 803E2D            <1>       cmp    byte [esi], '-'
3811 00009ECC 7799              <1>       ja     short pass_attr_space
3812 00009ECE E988000000        <1>         jmp     loc_attr_file_check_fname_fchar
3813                            <1>
3814                            <1> loc_chk_attr_enter:
3815 00009ED3 80FC0D            <1>       cmp    ah, 0Dh
3816 00009ED6 0F8585F0FFFF      <1>       jne    loc_cmd_failed
3817                            <1>
3818                            <1> pass_change_attr:
3819 00009EDC A0[443D0100]      <1>       mov    al, [Attr_Chars]
3820 00009EE1 F6D0              <1>       not    al
3821 00009EE3 2005[0C8C0100]    <1>       and    [Attributes], al
3822 00009EE9 A0[453D0100]      <1>       mov    al, [Attr_Chars+1]
3823 00009EEE 0805[0C8C0100]    <1>       or     [Attributes], al
3824                            <1>
3825                            <1> loc_show_attributes:
3826 00009EF4 BE[BF440100]      <1>       mov    esi, nextline
3827 00009EF9 E847D6FFFF        <1>       call   print_msg
3828                            <1>
3829                            <1> loc_show_attributes_no_nextline:
3830 00009EFE C705[443D0100]4E4F- <1>       mov    dword [Attr_Chars], 'NORM'
3830 00009F06 524D              <1>
3831 00009F08 66C705[483D0100]41- <1>       mov    word [Attr_Chars+4], 'AL'
3831 00009F10 4C                <1>
3832 00009F11 BE[443D0100]      <1>       mov    esi, Attr_Chars
3833 00009F16 A0[0C8C0100]      <1>       mov    al, [Attributes]
3834 00009F1B A804              <1>       test   al, 04h
3835 00009F1D 7406              <1>       jz     short pass_put_attr_s
3836 00009F1F 66C7065300        <1>       mov    word [esi], 0053h     ; S
3837 00009F24 46                <1>       inc    esi
3838                            <1>
3839                            <1> pass_put_attr_s:
3840 00009F25 A802              <1>       test   al, 02h
```

```
3841 00009F27 7406              <1>       jz      short pass_put_attr_h
3842 00009F29 66C7064800        <1>       mov     word [esi], 0048h     ; H
3843 00009F2E 46                <1>       inc     esi
3844                            <1>
3845                            <1> pass_put_attr_h:
3846 00009F2F A801              <1>       test    al, 01h
3847 00009F31 7406              <1>       jz      short pass_put_attr_r
3848 00009F33 66C7065200        <1>       mov     word [esi], 0052h     ; R
3849 00009F38 46                <1>       inc     esi
3850                            <1>
3851                            <1> pass_put_attr_r:
3852 00009F39 3C20              <1>       cmp     al, 20h
3853 00009F3B 7205              <1>       jb      short pass_put_attr_a
3854 00009F3D 66C7064100        <1>       mov     word [esi], 0041h     ; A
3855                            <1>
3856                            <1> pass_put_attr_a:
3857 00009F42 BE[373D0100]      <1>       mov     esi, Str_Attributes
3858 00009F47 E8F9D5FFFF        <1>       call    print_msg
3859 00009F4C BE[BF440100]      <1>       mov     esi, nextline
3860 00009F51 E8EFD5FFFF        <1>       call    print_msg
3861 00009F56 E944F9FFFF        <1>       jmp     loc_file_rw_restore_retn
3862                            <1>
3863                            <1> loc_attr_file_check_fname_fchar:
3864 00009F5B 46                <1>       inc     esi
3865 00009F5C 803E20            <1>       cmp     byte [esi], 20h
3866 00009F5F 74FA              <1>       je      short loc_attr_file_check_fname_fchar
3867 00009F61 0F8275FFFFFF      <1>         jb      pass_change_attr
3868                            <1>
3869                            <1> loc_attr_file_parse_path_name:
3870 00009F67 BF[F68A0100]      <1>       mov     edi, FindFile_Drv
3871 00009F6C E8D1160000        <1>       call    parse_path_name
3872 00009F71 0F82EAEFFFFF      <1>       jc      loc_cmd_failed
3873                            <1>
3874                            <1> loc_attr_file_check_filename_exists:
3875 00009F77 BE[388B0100]      <1>       mov     esi, FindFile_Name
3876 00009F7C 803E20            <1>       cmp     byte [esi], 20h
3877 00009F7F 0F86DCEFFFFF      <1>       jna     loc_cmd_failed
3878 00009F85 8935[B48B0100]    <1>       mov     [DelFile_FNPointer], esi
3879                            <1>
3880                            <1> loc_attr_file_drv:
3881 00009F8B 8A35[56820100]    <1>       mov     dh, [Current_Drv]
3882 00009F91 8835[B2890100]    <1>       mov     [RUN_CDRV], dh
3883                            <1>
3884 00009F97 8A15[F68A0100]    <1>       mov     dl, [FindFile_Drv]
3885 00009F9D 38F2              <1>       cmp     dl, dh
3886 00009F9F 740B              <1>       je      short loc_attr_file_change_directory
3887                            <1>
3888 00009FA1 E828E1FFFF        <1>       call    change_current_drive
3889 00009FA6 0F82F3F8FFFF      <1>       jc      loc_file_rw_cmd_failed
3890                            <1>
3891                            <1> loc_attr_file_change_directory:
3892 00009FAC 803D[F78A0100]20  <1>         cmp     byte [FindFile_Directory], 20h
3893 00009FB3 7618              <1>       jna     short loc_attr_file_find
3894                            <1>
3895 00009FB5 FE05[23380100]    <1>       inc     byte [Restore_CDIR]
3896                            <1>
3897 00009FBB BE[F78A0100]      <1>       mov     esi, FindFile_Directory
3898 00009FC0 30E4              <1>       xor     ah, ah ; CD_COMMAND sign -> 0
3899 00009FC2 E865100000        <1>       call    change_current_directory
3900 00009FC7 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 00009FCD 8B35[B48B0100]    <1>       mov     esi, [DelFile_FNPointer]
3908 00009FD3 66B80008          <1>       mov     ax, 0800h ; Except volume labels
3909 00009FD7 E895F4FFFF        <1>       call    find_first_file
3910 00009FDC 0F82BDF8FFFF      <1>       jc      loc_file_rw_cmd_failed
3911                            <1>
3912                            <1> loc_attr_file_ambgfn_check:
3913 00009FE2 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 00009FE5 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 00009FEB 66833D[443D0100]00 <1>       cmp     word [Attr_Chars], 0
3929 00009FF3 770B              <1>       ja      short loc_attr_file_change_attributes
3930 00009FF5 881D[0C8C0100]    <1>       mov     [Attributes], bl
3931 00009FFB E9F4FEFFFF        <1>       jmp     loc_show_attributes
3932                            <1>
3933                            <1> loc_attr_file_change_attributes:
3934 0000A000 A0[443D0100]      <1>       mov     al, [Attr_Chars]
3935 0000A005 F6D0              <1>       not     al
3936 0000A007 20C3              <1>       and     bl, al
3937 0000A009 A0[453D0100]      <1>       mov     al, [Attr_Chars+1]
3938 0000A00E 08C3              <1>       or      bl, al
3939                            <1>
3940 0000A010 66817F0CA101      <1>       cmp     word [edi+DirEntry_NTRes], 01A1h ; Singlix FS
3941 0000A016 741D              <1>       je      short loc_attr_file_fs_check
3942                            <1>
3943 0000A018 881D[0C8C0100]    <1>       mov     [Attributes], bl
3944 0000A01E 885F0B            <1>       mov     [edi+0Bh], bl    ; Attributes (New!)
3945                            <1>
```

```
                                <1>       ; 04/03/2016
                                <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 new attributes !
                                <1>
0000A021 C605[7C890100]02       <1>       mov   byte [DirBuff_ValidData], 2
                                <1>
0000A028 E8791A0000             <1>       call  save_directory_buffer
0000A02D 0F826CF8FFFF           <1>       jc    loc_file_rw_cmd_failed
                                <1>
0000A033 EB33                   <1>       jmp   short loc_print_attr_changed_message
                                <1>
                                <1> loc_attr_file_fs_check:
0000A035 29C0                   <1>       sub   eax, eax
0000A037 8A25[7A890100]         <1>         mov     ah, [DirBuff_DRV]
0000A03D BE00010900             <1>       mov   esi, Logical_DOSDisks
0000A042 01C6                   <1>         add     esi, eax
0000A044 807E04A1               <1>         cmp     byte [esi+LD_FSType], 0A1h
0000A048 7309                   <1>       jnc   short loc_attr_file_change_fs_file_attributes
                                <1>       ; 29/12/2017 (0Dh -> 29)
0000A04A 66B81D00               <1>       mov   ax, 29 ; Invalid Data
0000A04E E94CF8FFFF             <1>       jmp   loc_file_rw_cmd_failed
                                <1>
                                <1> loc_attr_file_change_fs_file_attributes:
                                <1>       ; BL = New MS-DOS File Attributes
0000A053 88D8                   <1>       mov   al, bl ; File/Directory Attributes
0000A055 30E4                   <1>       xor   ah, ah ; Attributes in MS-DOS format sign
0000A057 E873050000             <1>       call  change_fs_file_attributes
0000A05C 0F823DF8FFFF           <1>       jc    loc_file_rw_cmd_failed
                                <1>
0000A062 881D[0C8C0100]         <1>       mov   [Attributes], bl
                                <1>
                                <1> loc_print_attr_changed_message:
0000A068 BE[323D0100]           <1>       mov   esi, Msg_New
0000A06D E8D3D4FFFF             <1>       call  print_msg
0000A072 E987FEFFFF             <1>       jmp   loc_show_attributes_no_nextline
                                <1>
                                <1> rename_file:
                                <1>       ; 13/11/2017
                                <1>       ; 06/11/2016
                                <1>       ; 05/11/2016
                                <1>       ; 16/10/2016
                                <1>       ; 08/03/2016
                                <1>       ; 06/03/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>       ; 20/11/2010 (TRDOS v1, CMD_INTR.ASM, 'cmp_cmd_rename')
                                <1>       ; 16/11/2010
                                <1>
                                <1> get_rename_source_fchar:
                                <1>       ; esi = file name
0000A077 803E20                 <1>       cmp   byte [esi], 20h
0000A07A 7614                   <1>         jna  short loc_rename_nofilename_retn
                                <1>
0000A07C 8935[348C0100]         <1>       mov   [SourceFilePath], esi
                                <1>
                                <1> rename_scan_source_file:
0000A082 46                     <1>       inc   esi
0000A083 803E20                 <1>       cmp   byte [esi], 20h
0000A086 7409                   <1>       je    short rename_scan_destination_file_1
                                <1>       ;jb   short loc_rename_nofilename_retn
0000A088 0F82D3EEFFFF           <1>       jb    loc_cmd_failed
0000A08E EBF2                   <1>       jmp   short rename_scan_source_file
                                <1>
                                <1> loc_rename_nofilename_retn: ; 08/03/2016
0000A090 C3                     <1>       retn
                                <1>
                                <1> rename_scan_destination_file_1:
0000A091 C60600                 <1>       mov   byte [esi], 0
                                <1>
                                <1> rename_scan_destination_file_2:
0000A094 46                     <1>       inc   esi
0000A095 803E20                 <1>       cmp   byte [esi], 20h
0000A098 74FA                   <1>       je    short rename_scan_destination_file_2
                                <1>       ;jb   short loc_rename_nofilename_retn
0000A09A 0F82C1EEFFFF           <1>       jb    loc_cmd_failed
                                <1>
0000A0A0 8935[388C0100]         <1>       mov   [DestinationFilePath], esi
                                <1>
                                <1> rename_scan_destination_file_3:
0000A0A6 46                     <1>       inc   esi
0000A0A7 803E20                 <1>       cmp   byte [esi], 20h
0000A0AA 77FA                   <1>       ja    short rename_scan_destination_file_3
                                <1>
0000A0AC C60600                 <1>       mov   byte [esi], 0
                                <1>
                                <1> loc_rename_save_current_drive:
0000A0AF 8A35[56820100]         <1>       mov   dh, [Current_Drv]
0000A0B5 8835[B2890100]         <1>       mov   byte [RUN_CDRV], dh
                                <1>
                                <1> loc_rename_sf_parse_path_name:
0000A0BB 8B35[348C0100]         <1>       mov   esi, [SourceFilePath]
0000A0C1 BF[F68A0100]           <1>       mov   edi, FindFile_Drv
0000A0C6 E877150000             <1>       call  parse_path_name
0000A0CB 0F8290EEFFFF           <1>       jc    loc_cmd_failed
                                <1>
                                <1> loc_rename_sf_check_filename_exists:
0000A0D1 BE[388B0100]           <1>       mov   esi, FindFile_Name
0000A0D6 803E20                 <1>       cmp   byte [esi], 20h
0000A0D9 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>
```

```
4051 0000A0DF 8A15[F68A0100]      <1>       mov   dl, [FindFile_Drv]
4052 0000A0E5 38F2                <1>       cmp   dl, dh ; dh = [Current_Drv]
4053 0000A0E7 740B                <1>       je    short rename_sf_change_directory
4054                              <1>
4055 0000A0E9 E8E0DFFFFF          <1>       call  change_current_drive
4056 0000A0EE 0F82ABF7FFFF        <1>       jc    loc_file_rw_cmd_failed
4057                              <1>
4058                              <1> rename_sf_change_directory:
4059 0000A0F4 803D[F78A0100]20    <1>       cmp   byte [FindFile_Directory], 20h
4060 0000A0FB 7618                <1>       jna   short rename_sf_find
4061                              <1>
4062 0000A0FD FE05[23380100]      <1>       inc   byte [Restore_CDIR]
4063 0000A103 BE[F78A0100]        <1>       mov   esi, FindFile_Directory
4064 0000A108 30E4                <1>       xor   ah, ah ; CD_COMMAND sign -> 0
4065 0000A10A E81D0F0000          <1>       call  change_current_directory
4066 0000A10F 0F828AF7FFFF        <1>       jc    loc_file_rw_cmd_failed
4067                              <1>
4068                              <1> ;rename_sf_change_prompt_dir_string:
4069                              <1>       ;call  change_prompt_dir_string
4070                              <1>
4071                              <1> rename_sf_find:
4072                              <1>       ;mov   esi, [DelFile_FNPointer]
4073 0000A115 BE[388B0100]        <1>       mov   esi, FindFile_Name
4074                              <1>
4075 0000A11A 66B80008            <1>       mov   ax, 0800h ; Except volume labels
4076 0000A11E E84EF3FFFF          <1>       call  find_first_file
4077 0000A123 0F8276F7FFFF        <1>       jc    loc_file_rw_cmd_failed
4078                              <1>
4079                              <1> loc_rename_sf_ambgfn_check:
4080 0000A129 6621D2              <1>       and   dx, dx ; Ambiguous filename chars used sign (DX>0)
4081                              <1>       ;     (Note: It was BX in TRDOS v1)
4082                              <1>       ;jz   short loc_rename_sf_found
4083 0000A12C 0F8571FCFFFF        <1>       jnz   loc_file_not_found
4084                              <1>
4085                              <1>       ;mov  eax, 2 ; File not found sign
4086                              <1>       ;stc
4087                              <1>       ;jmp  loc_file_rw_cmd_failed
4088                              <1>
4089                              <1> loc_rename_sf_found:
4090                              <1>       ; EDI = Directory buffer entry offset/address
4091                              <1>       ; BL = File (or Directory) Attributes
4092                              <1>       ;     (Note: It was 'CL' in TRDOS v1)
4093                              <1>       ; mov  bl, [EDI+0Bh]
4094                              <1>
4095 0000A132 F6C307              <1>       test  bl, 07h ; Attributes, S-H-R
4096 0000A135 0F856EF7FFFF        <1>       jnz   loc_permission_denied
4097                              <1>
4098 0000A13B BE[F68A0100]        <1>         mov     esi, FindFile_Drv
4099 0000A140 BF[3C8C0100]        <1>         mov     edi, SourceFile_Drv
4100 0000A145 B920000000          <1>       mov   ecx, 32
4101 0000A14A F3A5                <1>       rep   movsd
4102                              <1>
4103                              <1> loc_rename_df_parse_path_name:
4104 0000A14C 8B35[388C0100]      <1>       mov   esi, [DestinationFilePath]
4105 0000A152 BF[F68A0100]        <1>       mov   edi, FindFile_Drv
4106 0000A157 E8E6140000          <1>       call  parse_path_name
4107 0000A15C 7219                <1>       jc    short loc_rename_df_cmd_failed
4108                              <1>
4109                              <1>       ;mov  dh, [RUN_CDRV]
4110 0000A15E 8A35[56820100]      <1>       mov   dh, [Current_Drv]
4111                              <1>
4112                              <1>       ; 'rename' command is valid only for same dos drive and same dir!
4113                              <1>       ; ('move' command must be used if source file and destination file
4114                              <1>       ; directories are not same!)
4115 0000A164 8A15[F68A0100]      <1>       mov   dl, [FindFile_Drv]
4116 0000A16A 38F2                <1>       cmp   dl, dh ; are source and destination drives different ?!
4117 0000A16C 7509                <1>       jne   short loc_rename_df_cmd_failed ; yes!
4118                              <1>
4119                              <1> rename_df_check_dirname_exists:
4120 0000A16E 803D[F78A0100]00    <1>       cmp   byte [FindFile_Directory], 0
4121 0000A175 760B                <1>       jna   short rename_df_check_filename_exists
4122                              <1>
4123                              <1>       ; different source file and destination file directories !
4124                              <1> loc_rename_df_cmd_failed:
4125 0000A177 B801000000          <1>       mov   eax, 1 ; TRDOS 'Bad command or file name' error
4126 0000A17C F9                  <1>       stc
4127 0000A17D E91DF7FFFF          <1>       jmp   loc_file_rw_cmd_failed
4128                              <1>
4129                              <1> rename_df_check_filename_exists:
4130 0000A182 BE[388B0100]        <1>       mov   esi, FindFile_Name
4131 0000A187 E8A3F6FFFF          <1>       call  check_filename
4132 0000A18C 0F82BFF7FFFF        <1>       jc    loc_mkdir_invalid_dir_name_chars
4133                              <1>
4134                              <1>       ;mov  [DelFile_FNPointer], esi
4135                              <1>       ;cmp  byte [esi], 20h
4136                              <1>       ;ja   short loc_rename_df_find
4137                              <1>
4138                              <1>       ;mov  dh, [Current_Drv] ; dh has not been changed
4139                              <1>
4140                              <1> rename_df_drv_check_writable:
4141 0000A192 0FB6F6              <1>       movzx  esi, dh
4142                              <1>       ;movzx esi, byte [Current_Drv]
4143 0000A195 81C600010900        <1>       add   esi, Logical_DOSDisks
4144                              <1>
4145 0000A19B 88F2                <1>       mov   dl, dh ; dl = [Current_Drv]
4146 0000A19D 8A7601              <1>       mov   dh, [esi+LD_DiskType]
4147                              <1>
4148 0000A1A0 80FE01              <1>       cmp   dh, 1 ; 0 = Invalid
4149 0000A1A3 7310                <1>       jnb   short rename_df_compare_sf_df_name
4150                              <1>
4151                              <1>       ; 16/10/2016 (13h -> 30)
4152 0000A1A5 B81E000000          <1>       mov   eax, 30 ; 'Disk write-protected' error
4153 0000A1AA 8B1D[388C0100]      <1>       mov   ebx, [DestinationFilePath]
4154 0000A1B0 E9EAF6FFFF          <1>       jmp   loc_file_rw_cmd_failed
4155                              <1>
```

```
                                <1> rename_df_compare_sf_df_name:
0000A1B5 BE[388B0100]           <1>       mov    esi, FindFile_Name
0000A1BA BF[7E8C0100]           <1>       mov    edi, SourceFile_Name
0000A1BF B90C000000             <1>       mov    ecx, 12
                                <1> rename_df_compare_sf_df_name_next:
0000A1C4 AC                     <1>       lodsb
0000A1C5 AE                     <1>       scasb
0000A1C6 7506                   <1>       jne    short loc_rename_df_find
0000A1C8 08C0                   <1>       or     al, al
0000A1CA 74AB                   <1>       jz     short loc_rename_df_cmd_failed
0000A1CC E2F6                   <1>       loop   rename_df_compare_sf_df_name_next
                                <1>
                                <1> loc_rename_df_find:
                                <1>       ;mov   esi, [DelFile_FNPointer]
0000A1CE BE[388B0100]           <1>       mov    esi, FindFile_Name
                                <1>
0000A1D3 6631C0                 <1>       xor    ax, ax ; Any
0000A1D6 E896F2FFFF             <1>       call   find_first_file
                                <1>       ;jnc   short loc_rename_df_found
                                <1>       ; 29/12/2017
0000A1DB 0F83C8F6FFFF           <1>       jnc    loc_permission_denied
                                <1>
                                <1> loc_rename_df_check_error_code:
                                <1>       ;cmp   eax, 2
0000A1E1 3C02                   <1>       cmp    al, 2 ; Not found error
0000A1E3 7406                   <1>       je     short rename_df_move_find_struct_to_dest
0000A1E5 F9                     <1>       stc
0000A1E6 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:
0000A1EB BE[F68A0100]           <1>         mov     esi, FindFile_Drv
0000A1F0 BF[BC8C0100]           <1>         mov     edi, DestinationFile_Drv
0000A1F5 B920000000             <1>       mov    ecx, 32
0000A1FA F3A5                   <1>       rep    movsd
                                <1>
                                <1> loc_rename_df_process_q_sf:
                                <1>       ;mov   ecx, 12
0000A1FC B10C                   <1>       mov    cl, 12
0000A1FE BE[7E8C0100]           <1>       mov    esi, SourceFile_Name
0000A203 BF[733D0100]           <1>       mov    edi, Rename_OldName
                                <1> rename_df_process_q_nml_1_sf:
0000A208 AC                     <1>       lodsb
0000A209 3C20                   <1>         cmp  al, 20h
0000A20B 7603                   <1>         jna  short rename_df_process_q_nml_2_sf
0000A20D AA                     <1>       stosb
0000A20E E2F8                   <1>       loop   rename_df_process_q_nml_1_sf
                                <1>
                                <1> rename_df_process_q_nml_2_sf:
0000A210 C60700                 <1>       mov    byte [edi], 0
                                <1>
                                <1> loc_rename_df_process_q_df:
                                <1>       ;mov   ecx, 12
0000A213 B10C                   <1>       mov    cl, 12
0000A215 BE[FE8C0100]           <1>       mov    esi, DestinationFile_Name
0000A21A BF[843D0100]           <1>       mov    edi, Rename_NewName
                                <1> rename_df_process_q_nml_1_df:
0000A21F AC                     <1>       lodsb
0000A220 3C20                   <1>       cmp    al, 20h
0000A222 7603                   <1>       jna    short loc_rename_df_process_q_nml_2_df
0000A224 AA                     <1>       stosb
0000A225 E2F8                   <1>       loop   rename_df_process_q_nml_1_df
                                <1>
                                <1> loc_rename_df_process_q_nml_2_df:
0000A227 C60700                 <1>       mov    byte [edi], 0
                                <1>
                                <1> loc_rename_confirmation_question:
0000A22A BE[4B3D0100]           <1>       mov    esi, Msg_DoYouWantRename
0000A22F E811D3FFFF             <1>       call   print_msg
                                <1>
0000A234 A0[998C0100]           <1>       mov    al, [SourceFile_DirEntry+11] ; Attributes
0000A239 2410                   <1>       and    al, 10h
0000A23B 750C                   <1>       jnz    short rename_confirmation_question_dir
                                <1>
                                <1> rename_confirmation_question_file:
0000A23D BE[623D0100]           <1>       mov    esi, Rename_File
0000A242 E8FED2FFFF             <1>       call   print_msg
0000A247 EB0A                   <1>       jmp    short rename_confirmation_question_as
                                <1>
                                <1> rename_confirmation_question_dir:
0000A249 BE[683D0100]           <1>       mov    esi, Rename_Directory
0000A24E E8F2D2FFFF             <1>       call   print_msg
                                <1>
                                <1> rename_confirmation_question_as:
0000A253 BE[733D0100]           <1>       mov    esi, Rename_OldName
0000A258 E8E8D2FFFF             <1>       call   print_msg
0000A25D BE[803D0100]           <1>       mov    esi, Msg_File_rename_as
0000A262 E8DED2FFFF             <1>       call   print_msg
0000A267 BE[A73C0100]           <1>       mov    esi, Msg_YesNo
0000A26C E8D4D2FFFF             <1>       call   print_msg
                                <1>
                                <1> loc_rename_ask_again:
0000A271 30E4                   <1>       xor    ah, ah
0000A273 E86D6CFFFF             <1>       call   int16h
0000A278 3C1B                   <1>       cmp    al, 1Bh
0000A27A 740F                   <1>       je     short loc_do_not_rename_file
0000A27C 24DF                   <1>       and    al, 0DFh
0000A27E A2[B13C0100]           <1>       mov    [Y_N_nextline], al
0000A283 3C59                   <1>       cmp    al, 'Y'
```

```
4261 0000A285 7404                <1>       je     short loc_yes_rename_file
4262 0000A287 3C4E                <1>       cmp    al, 'N'
4263 0000A289 75E6                <1>       jne    short loc_rename_ask_again
4264                              <1>
4265                              <1> loc_do_not_rename_file:
4266                              <1> loc_yes_rename_file:
4267 0000A28B E848F7FFFF          <1>       call   y_n_answer ; 29/12/2017
4268                              <1>       ;cmp   al, 'Y' ; 'yes'
4269                              <1>       ;cmc
4270                              <1>         ;jnc loc_file_rw_restore_retn
4271 0000A290 3C4E                <1>       cmp    al, 'N' ; 'no'
4272 0000A292 0F8407F6FFFF        <1>         je   loc_file_rw_restore_retn
4273                              <1>
4274 0000A298 BE[843D0100]        <1>       mov    esi, Rename_NewName
4275 0000A29D 668B0D[B68C0100]    <1>       mov    cx, [SourceFile_DirEntryNumber]
4276 0000A2A4 66A1[A28C0100]      <1>       mov    ax, [SourceFile_DirEntry+20] ; First Cluster, HW
4277 0000A2AA C1E010              <1>       shl    eax, 16 ; 13/11/2017
4278 0000A2AD 66A1[A88C0100]      <1>       mov    ax, [SourceFile_DirEntry+26] ; First Cluster, LW
4279                              <1>
4280 0000A2B3 0FB61D[8B8C0100]    <1>       movzx  ebx, byte [SourceFile_LongNameEntryLength]
4281 0000A2BA E8CB1B0000          <1>       call   rename_directory_entry
4282 0000A2BF E9FBF6FFFF          <1>       jmp    loc_rename_file_ok
4283                              <1> ;loc_rename_file_ok:
4284                              <1> ;     jc     loc_run_cmd_failed
4285                              <1> ;     mov    esi, Msg_OK
4286                              <1> ;     call   proc_printmsg
4287                              <1> ;     jmp    loc_file_rw_restore_retn
4288                              <1>
4289                              <1> move_file:
4290                              <1>       ; 11/03/2016
4291                              <1>       ; 09/03/2016
4292                              <1>       ; 08/03/2016 (TRDOS 386 = TRDOS v2.0)
4293                              <1>       ; 21/05/2011 (TRDOS v1, CMD_INTR.ASM, 'cmp_cmd_move')
4294                              <1>       ; 23/04/2011
4295                              <1>
4296                              <1> get_move_source_fchar:
4297                              <1>       ; esi = file name
4298 0000A2C4 803E20              <1>       cmp    byte [esi], 20h
4299 0000A2C7 7614                <1>         jna  short loc_move_nofilename_retn
4300                              <1>
4301 0000A2C9 8935[348C0100]      <1>       mov    [SourceFilePath], esi
4302                              <1>
4303                              <1> move_scan_source_file:
4304 0000A2CF 46                  <1>       inc    esi
4305 0000A2D0 803E20              <1>       cmp    byte [esi], 20h
4306 0000A2D3 7409                <1>         je      short move_scan_destination_1
4307                              <1>       ;jb    short loc_move_nofilename_retn
4308 0000A2D5 0F8286ECFFFF        <1>       jb     loc_cmd_failed
4309 0000A2DB EBF2                <1>       jmp    short move_scan_source_file
4310                              <1>
4311                              <1> loc_move_nofilename_retn:
4312 0000A2DD C3                  <1>       retn
4313                              <1>
4314                              <1> move_scan_destination_1:
4315 0000A2DE C60600              <1>       mov    byte [esi], 0
4316                              <1>
4317                              <1> move_scan_destination_2:
4318 0000A2E1 46                  <1>       inc    esi
4319 0000A2E2 803E20              <1>       cmp    byte [esi], 20h
4320 0000A2E5 74FA                <1>       je     short move_scan_destination_2
4321                              <1>       ;jb    short loc_move_nofilename_retn
4322 0000A2E7 0F8274ECFFFF        <1>       jb     loc_cmd_failed
4323                              <1>
4324 0000A2ED 8935[388C0100]      <1>       mov    [DestinationFilePath], esi
4325                              <1>
4326                              <1> move_scan_destination_3:
4327 0000A2F3 46                  <1>       inc    esi
4328 0000A2F4 803E20              <1>       cmp    byte [esi], 20h
4329 0000A2F7 77FA                <1>       ja     short move_scan_destination_3
4330 0000A2F9 C60600              <1>       mov    byte [esi], 0
4331                              <1>
4332                              <1> loc_move_scan_destination_OK:
4333 0000A2FC 8B35[348C0100]      <1>       mov    esi, [SourceFilePath]
4334 0000A302 8B3D[388C0100]      <1>       mov    edi, [DestinationFilePath]
4335                              <1>
4336 0000A308 B001                <1>       mov    al, 1  ; move procedure Phase 1
4337 0000A30A E8F71B0000          <1>       call   move_source_file_to_destination_file
4338 0000A30F 7328                <1>       jnc    short move_source_file_to_destination_question
4339                              <1>
4340                              <1> loc_move_cmd_failed_1:
4341 0000A311 08C0                <1>       or     al, al
4342 0000A313 0F8448ECFFFF        <1>       jz     loc_cmd_failed
4343 0000A319 3C11                <1>       cmp    al, 11h
4344 0000A31B 740D                <1>       je     short loc_msg_not_same_device
4345                              <1>       ;cmp   al, 05h
4346                              <1>       ;cmp   al, ERR_PERM_DENIED ; 29/12/2017
4347                              <1>       ;jne   loc_run_cmd_failed
4348                              <1>       ;jmp   loc_permission_denied
4349 0000A31D 3C0B                <1>       cmp    al, ERR_PERM_DENIED
4350 0000A31F 0F8484F5FFFF        <1>       je     loc_permission_denied
4351 0000A325 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 0000A32A BE[913D0100]        <1>       mov    esi, msg_not_same_drv
4359 0000A32F E811D2FFFF          <1>       call   print_msg
4360 0000A334 E966F5FFFF          <1>       jmp    loc_file_rw_restore_retn
4361                              <1>
4362                              <1> move_source_file_to_destination_question:
4363 0000A339 A0[3C8C0100]        <1>         mov     al, [SourceFile_Drv]
4364 0000A33E 0441                <1>       add    al, 'A'
4365 0000A340 A2[F33D0100]        <1>       mov    [msg_source_file_drv], al
```

```
4366 0000A345 A0[BC8C0100]        <1>         mov     al, [DestinationFile_Drv]
4367 0000A34A 0441                <1>       add   al, 'A'
4368 0000A34C A2[123E0100]        <1>       mov   [msg_destination_file_drv], al
4369                              <1>
4370 0000A351 57                  <1>       push  edi ; *
4371                              <1>
4372 0000A352 BE[D73D0100]        <1>       mov   esi, msg_source_file
4373 0000A357 E8E9D1FFFF          <1>       call  print_msg
4374 0000A35C BE[3D8C0100]        <1>       mov   esi, SourceFile_Directory
4375 0000A361 803E20              <1>       cmp   byte [esi], 20h
4376 0000A364 7605                <1>       jna   short msftdfq_sfn
4377 0000A366 E8DAD1FFFF          <1>       call  print_msg
4378                              <1> msftdfq_sfn:
4379 0000A36B BE[7E8C0100]        <1>       mov   esi, SourceFile_Name
4380 0000A370 E8D0D1FFFF          <1>       call  print_msg
4381 0000A375 BE[F63D0100]        <1>       mov   esi, msg_destination_file
4382 0000A37A E8C6D1FFFF          <1>       call  print_msg
4383 0000A37F BE[BD8C0100]        <1>       mov   esi, DestinationFile_Directory
4384 0000A384 803E20              <1>       cmp   byte [esi], 20h
4385 0000A387 7605                <1>       jna   short msftdfq_dfn
4386 0000A389 E8B7D1FFFF          <1>       call  print_msg
4387                              <1> msftdfq_dfn:
4388 0000A38E BE[FE8C0100]        <1>       mov   esi, DestinationFile_Name
4389 0000A393 E8ADD1FFFF          <1>       call  print_msg
4390 0000A398 BE[153E0100]        <1>       mov   esi, msg_copy_nextline
4391 0000A39D E8A3D1FFFF          <1>       call  print_msg
4392 0000A3A2 BE[153E0100]        <1>       mov   esi, msg_copy_nextline
4393 0000A3A7 E899D1FFFF          <1>       call  print_msg
4394                              <1>
4395                              <1> loc_move_ask_for_new_file_yes_no:
4396 0000A3AC BE[A33D0100]        <1>       mov   esi, Msg_DoYouWantMoveFile
4397 0000A3B1 E88FD1FFFF          <1>       call  print_msg
4398 0000A3B6 BE[A73C0100]        <1>       mov   esi, Msg_YesNo
4399 0000A3BB E885D1FFFF          <1>       call  print_msg
4400                              <1> loc_move_ask_for_new_file_again:
4401 0000A3C0 30E4                <1>       xor   ah, ah
4402 0000A3C2 E81E6BFFFF          <1>       call  int16h
4403 0000A3C7 3C1B                <1>       cmp   al, 1Bh
4404                              <1>       ;je   short loc_do_not_move_file
4405 0000A3C9 7441                <1>       je    short loc_move_y_n_escape
4406 0000A3CB 24DF                <1>       and   al, 0DFh
4407 0000A3CD A2[B13C0100]        <1>         mov     [Y_N_nextline], al
4408 0000A3D2 3C59                <1>       cmp   al, 'Y'
4409 0000A3D4 7404                <1>       je    short loc_yes_move_file
4410 0000A3D6 3C4E                <1>       cmp   al, 'N'
4411 0000A3D8 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 0000A3DA E8F9F5FFFF          <1>       call  y_n_answer ; 29/12/2017
4416 0000A3DF 5F                  <1>       pop   edi ; *
4417                              <1>       ;cmp  al, 'Y' ; 'yes'
4418                              <1>       ;cmc
4419                              <1>         ;jnc loc_file_rw_restore_retn
4420 0000A3E0 3C4E                <1>       cmp   al, 'N' ; 'no'
4421 0000A3E2 0F84B7F4FFFF        <1>         je  loc_file_rw_restore_retn
4422                              <1>
4423                              <1> loc_move_yes_move_file:
4424 0000A3E8 B002                <1>       mov   al, 2 ; move procedure Phase 2
4425 0000A3EA E8171B0000          <1>       call  move_source_file_to_destination_file
4426                              <1>       ;jc   short loc_move_cmd_failed_2
4427 0000A3EF 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 0000A3F5 3C27                <1>       cmp   al, 27h
4436 0000A3F7 0F858FEBFFFF        <1>       jne   loc_run_cmd_failed
4437                              <1>
4438 0000A3FD BE[BC3D0100]        <1>       mov   esi, msg_insufficient_disk_space
4439 0000A402 E83ED1FFFF          <1>       call  print_msg
4440                              <1>
4441 0000A407 E993F4FFFF          <1>       jmp   loc_file_rw_restore_retn
4442                              <1>
4443                              <1> loc_move_y_n_escape:
4444 0000A40C B04E                <1>       mov   al, 'N' ; 'no'
4445 0000A40E 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
4454                              <1>
4455                              <1> get_copy_source_fchar:
4456                              <1>       ; esi = file name
4457 0000A410 803E20              <1>       cmp   byte [esi], 20h
4458 0000A413 7614                <1>         jna     short loc_copy_nofilename_retn
4459                              <1>
4460 0000A415 8935[348C0100]      <1>       mov   [SourceFilePath], esi
4461                              <1>
4462                              <1> copy_scan_source_file:
4463 0000A41B 46                  <1>       inc   esi
4464 0000A41C 803E20              <1>       cmp   byte [esi], 20h
4465 0000A41F 7409                <1>       je    short copy_scan_destination_1
4466                              <1>       ;jb   short loc_copy_nofilename_retn
4467 0000A421 0F823AEBFFFF        <1>       jb    loc_cmd_failed
4468 0000A427 EBF2                <1>       jmp   short copy_scan_source_file
4469                              <1>
4470                              <1> loc_copy_nofilename_retn:
```

```
4471 0000A429 C3                <1>       retn
4472                            <1>
4473                            <1> copy_scan_destination_1:
4474 0000A42A C60600            <1>       mov    byte [esi], 0
4475                            <1>
4476                            <1> copy_scan_destination_2:
4477 0000A42D 46                <1>       inc    esi
4478 0000A42E 803E20            <1>       cmp    byte [esi], 20h
4479 0000A431 74FA              <1>       je     short copy_scan_destination_2
4480                            <1>       ;jb    short loc_copy_nofilename_retn
4481 0000A433 0F8228EBFFFF      <1>       jb     loc_cmd_failed
4482                            <1>
4483 0000A439 8935[388C0100]    <1>       mov    [DestinationFilePath], esi
4484                            <1>
4485                            <1> copy_scan_destination_3:
4486 0000A43F 46                <1>       inc    esi
4487 0000A440 803E20            <1>       cmp    byte [esi], 20h
4488 0000A443 77FA              <1>       ja     short copy_scan_destination_3
4489 0000A445 C60600            <1>       mov    byte [esi], 0
4490                            <1>
4491                            <1> loc_copy_save_current_drive:
4492 0000A448 8A35[56820100]    <1>       mov    dh, [Current_Drv]
4493 0000A44E 8835[B2890100]    <1>       mov    [RUN_CDRV], dh
4494                            <1>
4495                            <1> copy_source_file_to_destination_phase_1:
4496 0000A454 8B35[348C0100]    <1>       mov    esi, [SourceFilePath]
4497 0000A45A 8B3D[388C0100]    <1>       mov    edi, [DestinationFilePath]
4498                            <1>
4499 0000A460 B001              <1>       mov    al, 1  ; copy procedure Phase 1
4500 0000A462 E83C1D0000        <1>       call   copy_source_file_to_destination_file
4501 0000A467 732B              <1>       jnc    short copy_source_file_to_destination_question
4502                            <1>
4503                            <1> loc_copy_cmd_failed_1:
4504                            <1>       ; 18/03/2016 (restore current drive and directory)
4505 0000A469 08C0              <1>       or     al, al
4506 0000A46B 7507              <1>       jnz    short loc_copy_cmd_failed_2
4507                            <1>
4508 0000A46D FEC0              <1>         inc     al ; mov al, 1 ; Bad command or file name !
4509 0000A46F E918EBFFFF        <1>       jmp    loc_run_cmd_failed
4510                            <1>
4511                            <1> loc_copy_cmd_failed_2:
4512 0000A474 3C27              <1>       cmp    al, 27h ; Insufficient disk space
4513 0000A476 740D              <1>       je     short loc_file_write_insuff_disk_space_msg
4514                            <1>
4515                            <1>       ; 29/12/2017
4516                            <1>       ;cmp   al, 05h
4517 0000A478 3C0B              <1>       cmp    al, ERR_PERM_DENIED
4518 0000A47A 0F850CEBFFFF      <1>       jne    loc_run_cmd_failed
4519                            <1>
4520 0000A480 E924F4FFFF        <1>       jmp    loc_permission_denied
4521                            <1>
4522                            <1> loc_file_write_insuff_disk_space_msg:
4523 0000A485 BE[BC3D0100]      <1>       mov    esi, msg_insufficient_disk_space
4524 0000A48A E8B6D0FFFF        <1>       call   print_msg
4525 0000A48F E90BF4FFFF        <1>         jmp     loc_file_rw_restore_retn
4526                            <1>
4527                            <1> copy_source_file_to_destination_question:
4528 0000A494 57                <1>       push   edi ; *
4529                            <1>
4530                            <1>       ; dh = source file attributes
4531                            <1>       ; dl > 0 -> destination file found
4532 0000A495 20D2              <1>       and    dl, dl
4533 0000A497 7449              <1>       jz     short copy_source_file_to_destination_pass_owrq
4534                            <1>
4535                            <1> loc_copy_ask_for_owr_yes_no:
4536 0000A499 BE[183E0100]      <1>       mov    esi, Msg_DoYouWantOverWriteFile
4537 0000A49E E8A2D0FFFF        <1>       call   print_msg
4538 0000A4A3 BE[FE8C0100]      <1>       mov    esi, DestinationFile_Name
4539 0000A4A8 E898D0FFFF        <1>       call   print_msg
4540 0000A4AD BE[A73C0100]      <1>       mov    esi, Msg_YesNo
4541 0000A4B2 E88ED0FFFF        <1>       call   print_msg
4542                            <1>
4543                            <1> loc_copy_ask_for_owr_again:
4544 0000A4B7 30E4              <1>       xor    ah, ah
4545 0000A4B9 E8276AFFFF        <1>       call   int16h
4546 0000A4BE 3C1B              <1>       cmp    al, 1Bh
4547                            <1>         ;je     loc_do_not_copy_file
4548 0000A4C0 7419              <1>         je      short loc_copy_y_n_escape
4549 0000A4C2 24DF              <1>       and    al, 0DFh
4550 0000A4C4 A2[B13C0100]      <1>         mov     [Y_N_nextline], al
4551 0000A4C9 3C59              <1>       cmp    al, 'Y'
4552 0000A4CB 0F84B1000000      <1>         je      loc_yes_copy_file
4553 0000A4D1 3C4E              <1>       cmp    al, 'N'
4554 0000A4D3 0F84A9000000      <1>         je      loc_do_not_copy_file
4555 0000A4D9 EBDC              <1>       jmp    short loc_copy_ask_for_owr_again
4556                            <1>
4557                            <1> loc_copy_y_n_escape:
4558 0000A4DB B04E              <1>       mov    al, 'N' ; 'no'
4559 0000A4DD E9A0000000        <1>         jmp     loc_do_not_copy_file
4560                            <1>
4561                            <1> copy_source_file_to_destination_pass_owrq:
4562 0000A4E2 A0[3C8C0100]      <1>       mov     al, [SourceFile_Drv]
4563 0000A4E7 0441              <1>       add    al, 'A'
4564 0000A4E9 A2[F33D0100]      <1>       mov    [msg_source_file_drv], al
4565 0000A4EE A0[BC8C0100]      <1>         mov     al, [DestinationFile_Drv]
4566 0000A4F3 0441              <1>       add    al, 'A'
4567 0000A4F5 A2[123E0100]      <1>       mov    [msg_destination_file_drv], al
4568                            <1>
4569 0000A4FA BE[D73D0100]      <1>       mov    esi, msg_source_file
4570 0000A4FF E841D0FFFF        <1>       call   print_msg
4571 0000A504 BE[3D8C0100]      <1>       mov    esi, SourceFile_Directory
4572 0000A509 803E20            <1>       cmp    byte [esi], 20h
4573 0000A50C 7605              <1>       jna    short csftdfq_sfn
4574 0000A50E E832D0FFFF        <1>       call   print_msg
4575                            <1> csftdfq_sfn:
```

```
4576 0000A513 BE[7E8C0100]        <1>       mov   esi, SourceFile_Name
4577 0000A518 E828D0FFFF          <1>       call  print_msg
4578 0000A51D BE[F63D0100]        <1>       mov   esi, msg_destination_file
4579 0000A522 E81ED0FFFF          <1>       call  print_msg
4580 0000A527 BE[BD8C0100]        <1>       mov   esi, DestinationFile_Directory
4581 0000A52C 803E20              <1>       cmp   byte [esi], 20h
4582 0000A52F 7605                <1>       jna   short csftdfq_dfn
4583 0000A531 E80FD0FFFF          <1>       call  print_msg
4584                              <1> csftdfq_dfn:
4585 0000A536 BE[FE8C0100]        <1>       mov   esi, DestinationFile_Name
4586 0000A53B E805D0FFFF          <1>       call  print_msg
4587 0000A540 BE[153E0100]        <1>       mov   esi, msg_copy_nextline
4588 0000A545 E8FBCFFFFF          <1>       call  print_msg
4589 0000A54A BE[153E0100]        <1>       mov   esi, msg_copy_nextline
4590 0000A54F E8F1CFFFFF          <1>       call  print_msg
4591                              <1>
4592                              <1> loc_copy_ask_for_new_file_yes_no:
4593 0000A554 BE[373E0100]        <1>       mov   esi, Msg_DoYouWantCopyFile
4594 0000A559 E8E7CFFFFF          <1>       call  print_msg
4595 0000A55E BE[A73C0100]        <1>       mov   esi, Msg_YesNo
4596 0000A563 E8DDCFFFFF          <1>       call  print_msg
4597                              <1>
4598                              <1> loc_copy_ask_for_new_file_again:
4599 0000A568 30E4                <1>       xor   ah, ah
4600 0000A56A E87669FFFF          <1>       call  int16h
4601 0000A56F 3C1B                <1>       cmp   al, 1Bh
4602 0000A571 740F                <1>       je    short loc_do_not_copy_file
4603 0000A573 24DF                <1>       and   al, 0DFh
4604 0000A575 A2[B13C0100]        <1>         mov     [Y_N_nextline], al
4605 0000A57A 3C59                <1>       cmp   al, 'Y'
4606 0000A57C 7404                <1>       je    short loc_yes_copy_file
4607 0000A57E 3C4E                <1>       cmp   al, 'N'
4608 0000A580 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 0000A582 E851F4FFFF          <1>       call  y_n_answer ; 29/12/2017
4613 0000A587 5F                  <1>       pop   edi ; *
4614                              <1>       ;cmp  al, 'Y' ; 'yes'
4615                              <1>       ;cmc
4616                              <1>         ;jnc loc_file_rw_restore_retn
4617 0000A588 3C4E                <1>       cmp   al, 'N' ; 'no'
4618 0000A58A 0F840FF3FFFF        <1>         je  loc_file_rw_restore_retn
4619                              <1>
4620                              <1> copy_source_file_to_destination_pass_q:
4621 0000A590 B002                <1>       mov   al, 2  ; copy procedure Phase 2
4622 0000A592 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 0000A597 51                  <1>       push  ecx ; 29/12/2017
4628 0000A598 BE[153E0100]        <1>       mov   esi, msg_copy_nextline
4629 0000A59D E8A3CFFFFF          <1>       call  print_msg
4630 0000A5A2 58                  <1>       pop   eax ; 29/12/2017
4631                              <1>       ;;pop cx
4632                              <1>       ;pop  ax
4633                              <1>
4634                              <1>       ;or   cl, cl
4635 0000A5A3 08C0                <1>       or    al, al
4636 0000A5A5 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 0000A5A7 3C12                <1>       cmp   al, 18
4642 0000A5A9 7506                <1>       jne   short copy_source_file_to_destination_not_OK
4643                              <1>       ;
4644                              <1>       ;mov  al, cl ; error number (write fault!)
4645 0000A5AB F9                  <1>       stc
4646 0000A5AC E9EEF2FFFF          <1>       jmp   loc_file_rw_cmd_failed
4647                              <1>
4648                              <1> copy_source_file_to_destination_not_OK:
4649 0000A5B1 BE[503E0100]        <1>       mov   esi, Msg_read_file_error_before_EOF
4650 0000A5B6 E88ACFFFFF          <1>       call  print_msg
4651 0000A5BB E9DFF2FFFF          <1>       jmp   loc_file_rw_restore_retn
4652                              <1>
4653                              <1> copy_source_file_to_destination_OK:
4654 0000A5C0 BE[B53C0100]        <1>       mov   esi, Msg_OK
4655 0000A5C5 E87BCFFFFF          <1>       call  print_msg
4656                              <1>
4657 0000A5CA 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 0000A5CF 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')
```

```
                                <1>       ; 2005 - 28/08/2011
                                <1> get_setenv_fchar:
                                <1>       ; esi = environment variable/string
0000A5D0 8A06                   <1>       mov   al, [esi]
0000A5D2 3C20                   <1>       cmp   al, 20h
0000A5D4 771E                   <1>       ja    short loc_find_env
                                <1>
0000A5D6 BE00300900             <1>       mov   esi, Env_Page
                                <1> loc_print_setline:
0000A5DB 803E00                 <1>       cmp   byte [esi], 0
0000A5DE 7613                   <1>       jna   short loc_setenv_retn
0000A5E0 E860CFFFFF             <1>       call  print_msg
0000A5E5 56                     <1>       push  esi
0000A5E6 BE[BF440100]           <1>       mov   esi, nextline
0000A5EB E855CFFFFF             <1>       call  print_msg
0000A5F0 5E                     <1>       pop   esi
0000A5F1 EBE8                   <1>       jmp   short loc_print_setline
                                <1>
                                <1> loc_setenv_retn:
0000A5F3 C3                     <1>       retn
                                <1>
                                <1> loc_find_env:
0000A5F4 3C3D                   <1>       cmp   al, '='
0000A5F6 0F8465E9FFFF           <1>       je    loc_cmd_failed
                                <1>
0000A5FC 56                     <1>       push  esi
                                <1> loc_repeat_env_equal_check:
0000A5FD 46                     <1>       inc   esi
0000A5FE 803E3D                 <1>       cmp   byte [esi], '='
0000A601 7431                   <1>       je    short pass_env_equal_check
0000A603 803E20                 <1>       cmp   byte [esi], 20h
0000A606 73F5                   <1>       jnb   short loc_repeat_env_equal_check
0000A608 C60600                 <1>       mov   byte [esi], 0
0000A60B 5E                     <1>       pop   esi
0000A60C BF[56830100]           <1>       mov   edi, TextBuffer ; out buffer
0000A611 B9FF000000             <1>       mov   ecx, 255 ; maximum size (limit)
0000A616 30C0                   <1>       xor   al, al ; 0 -> use [ESI]
0000A618 E89E000000             <1>       call  get_environment_string
0000A61D 72D4                   <1>       jc    short loc_setenv_retn
                                <1>
0000A61F BE[56830100]           <1>       mov   esi, TextBuffer
0000A624 E81CCFFFFF             <1>       call  print_msg
0000A629 BE[BF440100]           <1>       mov   esi, nextline
0000A62E E812CFFFFF             <1>       call  print_msg
                                <1>
0000A633 C3                     <1>       retn
                                <1>
                                <1> pass_env_equal_check:
0000A634 46                     <1>       inc   esi
0000A635 803E20                 <1>       cmp   byte [esi], 20h
0000A638 73FA                   <1>       jnb   short pass_env_equal_check
0000A63A C60600                 <1>       mov   byte [esi], 0
                                <1>
                                <1> loc_call_set_env_string:
0000A63D 5E                     <1>       pop   esi
0000A63E E83B010000             <1>       call  set_environment_string
0000A643 73AE                   <1>       jnc   short loc_setenv_retn
                                <1>
                                <1> loc_set_cmd_failed:
0000A645 3C08                   <1>       cmp   al, 08h
0000A647 0F8514E9FFFF           <1>       jne   loc_cmd_failed
                                <1>
0000A64D BE[903E0100]           <1>       mov   esi, Msg_No_Set_Space
0000A652 E8EECEFFFF             <1>       call  print_msg
                                <1>
0000A657 C3                     <1>       retn
                                <1>
                                <1> set_get_path:
                                <1>       ; 11/04/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>       ; 03/09/2011 (TRDOS v1, CMD_INTR.ASM, 'cmp_cmd_path')
                                <1>       ; 2005
                                <1> get_path_fchar:
                                <1>       ; esi = path
0000A658 803E20                 <1>       cmp   byte [esi], 20h
0000A65B 7737                   <1>       ja    short loc_set_path
                                <1>
0000A65D BE00300900             <1>       mov   esi, Env_Page
                                <1> loc_print_path:
0000A662 803E00                 <1>       cmp   byte [esi], 0
0000A665 762C                   <1>       jna   short loc_path_retn
                                <1>
0000A667 BE[EF380100]           <1>       mov   esi, Cmd_Path ; 'PATH' address
0000A66C BF[56830100]           <1>       mov   edi, TextBuffer ; out buffer
0000A671 30C0                   <1>       xor   al, al  ; use [ESI]
0000A673 B9FF000000             <1>       mov   ecx, 255 ; maximum size (limit)
0000A678 E83E000000             <1>       call  get_environment_string
0000A67D 7214                   <1>       jc    short loc_path_retn
                                <1>
0000A67F BE[56830100]           <1>       mov   esi, TextBuffer
0000A684 E8BCCEFFFF             <1>       call  print_msg
0000A689 BE[BF440100]           <1>       mov   esi, nextline
0000A68E E8B2CEFFFF             <1>       call  print_msg
                                <1>
                                <1> loc_path_retn:
0000A693 C3                     <1>       retn
                                <1>
                                <1> loc_set_path:
0000A694 56                     <1>       push  esi
                                <1> loc_set_path_find_end:
0000A695 46                     <1>       inc   esi
0000A696 803E20                 <1>       cmp   byte [esi], 20h
0000A699 73FA                   <1>       jnb   short loc_set_path_find_end
0000A69B C60600                 <1>       mov   byte [esi], 0
                                <1> loc_set_path_header:
0000A69E 5E                     <1>       pop   esi
```

```
4786                                <1> set_path_x: ; 31/12/2017 ('syspath')
4787 0000A69F 4E                    <1>         dec    esi
4788 0000A6A0 C6063D                <1>         mov    byte [esi], '='
4789 0000A6A3 4E                    <1>         dec    esi
4790 0000A6A4 C60648                <1>         mov    byte [esi], 'H'
4791 0000A6A7 4E                    <1>         dec    esi
4792 0000A6A8 C60654                <1>         mov    byte [esi], 'T'
4793 0000A6AB 4E                    <1>         dec    esi
4794 0000A6AC C60641                <1>         mov    byte [esi], 'A'
4795 0000A6AF 4E                    <1>         dec    esi
4796 0000A6B0 C60650                <1>         mov    byte [esi], 'P'
4797                                <1>
4798                                <1> loc_path_call_set_env_string:
4799 0000A6B3 E8C6000000            <1>         call   set_environment_string
4800 0000A6B8 728B                  <1>           jc   short loc_set_cmd_failed
4801                                <1>
4802 0000A6BA 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 0000A6BB BA00300900            <1>         mov    edx, Env_Page
4827 0000A6C0 803A00                <1>         cmp    byte [edx], 0
4828 0000A6C3 7474                  <1>         jz     short get_env_string_with_word_stc_retn
4829                                <1>
4830 0000A6C5 66890D[C08D0100]      <1>         mov    [env_var_length], cx
4831                                <1>
4832 0000A6CC 51                    <1>         push   ecx ; *
4833 0000A6CD 56                    <1>         push   esi ; **
4834                                <1>
4835 0000A6CE 08C0                  <1>         or     al, al
4836 0000A6D0 7449                  <1>         jz     short get_env_string_with_word
4837                                <1>
4838                                <1> get_env_string_with_seq_number:
4839 0000A6D2 B101                  <1>         mov    cl, 1
4840 0000A6D4 88C5                  <1>         mov    ch, al
4841 0000A6D6 31C0                  <1>         xor    eax, eax
4842 0000A6D8 89D6                  <1>         mov    esi, edx ; Env_Page
4843                                <1>
4844                                <1> get_env_string_seq_number_check:
4845 0000A6DA 38CD                  <1>         cmp    ch, cl
4846 0000A6DC 7726                  <1>         ja     short get_env_string_seq_number_next
4847                                <1>
4848                                <1> get_env_string_move_to_buff:
4849 0000A6DE 57                    <1>         push   edi ; ***
4850                                <1>
4851 0000A6DF 29D2                  <1>         sub    edx, edx
4852                                <1>
4853                                <1> get_env_string_seq_number_repeat1:
4854 0000A6E1 42                    <1>         inc    edx
4855 0000A6E2 AC                    <1>         lodsb
4856 0000A6E3 AA                    <1>         stosb
4857                                <1>
4858 0000A6E4 66FF0D[C08D0100]      <1>         dec    word [env_var_length]
4859 0000A6EB 7508                  <1>         jnz    short get_env_string_seq_number_repeat3
4860                                <1>
4861                                <1> get_env_string_seq_number_repeat2:
4862 0000A6ED 20C0                  <1>         and    al, al
4863 0000A6EF 7408                  <1>         jz     short get_env_string_seq_number_ok
4864 0000A6F1 42                    <1>         inc    edx
4865 0000A6F2 AC                    <1>         lodsb
4866 0000A6F3 EBF8                  <1>         jmp    short get_env_string_seq_number_repeat2
4867                                <1>
4868                                <1> get_env_string_seq_number_repeat3:
4869 0000A6F5 08C0                  <1>         or     al, al
4870 0000A6F7 75E8                  <1>         jnz    short get_env_string_seq_number_repeat1
4871                                <1>
4872                                <1> get_env_string_seq_number_ok:
4873 0000A6F9 5F                    <1>         pop    edi ; ***
4874 0000A6FA 89D0                  <1>         mov    eax, edx ; Length of the environment string
4875                                <1>                         ; (ASCIIZ, includes ZERO tail)
4876 0000A6FC BA00300900            <1>         mov    edx, Env_Page
4877                                <1>
4878                                <1> get_env_string_stc_retn:
4879 0000A701 5E                    <1>         pop    esi ; **
4880 0000A702 59                    <1>         pop    ecx ; *
4881 0000A703 C3                    <1>         retn
4882                                <1>
4883                                <1> get_env_string_seq_number_next:
4884 0000A704 AC                    <1>         lodsb
4885 0000A705 08C0                  <1>         or     al, al
4886 0000A707 75FB                  <1>         jnz    short get_env_string_seq_number_next
4887                                <1>
4888 0000A709 81FE00320900          <1>         cmp    esi, Env_Page + Env_Page_Size ; +512 (+4096)
4889 0000A70F F5                    <1>         cmc
4890 0000A710 72EF                  <1>         jc     short get_env_string_stc_retn
```

```
                                <1>
0000A712 AC                     <1>       lodsb
0000A713 3C01                   <1>       cmp    al, 1
0000A715 72EA                   <1>       jb     short get_env_string_stc_retn
0000A717 FEC1                   <1>       inc    cl
0000A719 EBBF                   <1>       jmp    short get_env_string_seq_number_check
                                <1>
                                <1> get_env_string_with_word:
0000A71B 31C9                   <1>       xor    ecx, ecx
                                <1>
                                <1> get_env_string_calc_word_length:
0000A71D AC                     <1>       lodsb
0000A71E 3C20                   <1>       cmp    al, 20h
0000A720 7211                   <1>       jb     short get_env_string_calc_word_length_ok
                                <1>       ;inc   cx
0000A722 FEC1                   <1>       inc    cl
                                <1>
0000A724 3C61                   <1>       cmp    al, 'a'
0000A726 72F5                   <1>       jb     short get_env_string_calc_word_length
0000A728 3C7A                   <1>       cmp    al, 'z'
0000A72A 77F1                   <1>       ja     short get_env_string_calc_word_length
0000A72C 24DF                   <1>       and    al, 0DFh
0000A72E 8846FF                 <1>       mov    [esi-1], al
0000A731 EBEA                   <1>       jmp    short get_env_string_calc_word_length
                                <1>
                                <1> get_env_string_calc_word_length_ok:
0000A733 08C9                   <1>       or     cl, cl
0000A735 7506                   <1>       jnz    short get_env_string_calc_word_length_save
                                <1>
0000A737 5E                     <1>       pop    esi ; **
                                <1>
                                <1> get_env_string_stc_retn1:
0000A738 59                     <1>       pop    ecx ; *
                                <1>
                                <1> get_env_string_with_word_stc_retn:
0000A739 31C0                   <1>       xor    eax, eax
0000A73B F9                     <1>       stc
0000A73C C3                     <1>       retn
                                <1>
                                <1> get_env_string_calc_word_length_save:
0000A73D 871C24                 <1>       xchg   ebx, [esp] ; **
0000A740 89DE                   <1>       mov    esi, ebx
                                <1>              ; Start of the env string (to be searched)
                                <1>
0000A742 57                     <1>       push   edi ; ***
0000A743 89D7                   <1>       mov    edi, edx ; Env_Page
                                <1>
                                <1> get_env_string_compare:
0000A745 57                     <1>       push   edi ; ****
0000A746 51                     <1>       push   ecx ; ***** ; Variable name length
                                <1>
                                <1> get_env_string_compare_rep:
0000A747 AC                     <1>       lodsb
0000A748 AE                     <1>       scasb
0000A749 7511                   <1>       jne    short get_env_string_compare_next1
0000A74B E2FA                   <1>       loop   get_env_string_compare_rep
                                <1>
0000A74D 803F3D                 <1>       cmp    byte [edi], '='
0000A750 750A                   <1>       jne    short get_env_string_compare_next1
                                <1>
0000A752 59                     <1>       pop    ecx ; *****
0000A753 5F                     <1>       pop    edi ; ****
0000A754 89FE                   <1>       mov    esi, edi
0000A756 5F                     <1>       pop    edi ; ***
0000A757 871C24                 <1>       xchg   ebx, [esp] ; **
0000A75A EB82                   <1>       jmp    short get_env_string_move_to_buff
                                <1>
                                <1> get_env_string_compare_next1:
0000A75C 89FE                   <1>       mov    esi, edi
0000A75E 59                     <1>       pop    ecx ; *****
0000A75F 5F                     <1>       pop    edi ; ****
                                <1> get_env_string_compare_next2:
0000A760 81FEFF310900           <1>       cmp    esi, Env_Page + Env_Page_Size - 1 ; +511 (+4095)
0000A766 7310                   <1>       jnb    short get_env_string_compare_not_ok
0000A768 20C0                   <1>       and    al, al
0000A76A AC                     <1>       lodsb
0000A76B 75F3                   <1>       jnz    short get_env_string_compare_next2
0000A76D 08C0                   <1>       or     al, al
0000A76F 7407                   <1>       jz     short get_env_string_compare_not_ok
0000A771 4E                     <1>       dec    esi ; 12/04/2016
0000A772 89F7                   <1>       mov    edi, esi
0000A774 89DE                   <1>       mov    esi, ebx
0000A776 EBCD                   <1>       jmp    short get_env_string_compare
                                <1>
                                <1> get_env_string_compare_not_ok:
0000A778 5F                     <1>       pop    edi ; ***
0000A779 89DE                   <1>       mov    esi, ebx
0000A77B 5B                     <1>       pop    ebx ; **
0000A77C EBBA                   <1>       jmp    short get_env_string_stc_retn1
                                <1>
                                <1> set_environment_string:
                                <1>       ; 13/04/2016
                                <1>       ; 12/04/2016
                                <1>       ; 11/04/2016
                                <1>       ; 06/04/2016
                                <1>       ; 05/04/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>       ; 02/09/2011 (TRDOS v1, MAINPROG.ASM)
                                <1>       ; 29/08/2011
                                <1>       ; 29/08/2011
                                <1>       ; INPUT->
                                <1>       ;      ESI = ASCIIZ environment string
                                <1>       ; OUTPUT ->
                                <1>       ;      ESI is not changed
                                <1>       ;      CF = 1 -> Could not set,
                                <1>       ;           insufficient environment space
```

```
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 0000A77E 56                        <1>       push   esi ; *
5003                                    <1>
5004 0000A77F 31C0                      <1>       xor    eax, eax
5005                                    <1>
5006                                    <1> set_env_chk_validation1:
5007 0000A781 FEC4                      <1>       inc    ah ; variable (string) length
5008 0000A783 AC                        <1>       lodsb
5009 0000A784 3C3D                      <1>       cmp    al, '='
5010 0000A786 7415                      <1>       je     short set_env_chk_validation2
5011 0000A788 3C20                      <1>       cmp    al, 20h
5012 0000A78A 720F                      <1>       jb     short set_env_string_stc
5013                                    <1>
5014                                    <1>       ; 06/04/2016
5015 0000A78C 3C61                      <1>       cmp    al, 'a'
5016 0000A78E 72F1                      <1>       jb     short set_env_chk_validation1
5017 0000A790 3C7A                      <1>       cmp    al, 'z'
5018 0000A792 77ED                      <1>       ja     short set_env_chk_validation1
5019 0000A794 2C20                      <1>       sub    al, 'a'-'A'
5020 0000A796 8846FF                    <1>       mov    [esi-1], al
5021 0000A799 EBE6                      <1>       jmp    short set_env_chk_validation1
5022                                    <1>
5023                                    <1> set_env_string_stc:
5024 0000A79B 5E                        <1>       pop    esi ; *
5025                                    <1>       ;stc
5026 0000A79C C3                        <1>       retn
5027                                    <1>
5028                                    <1> set_env_chk_validation2:
5029 0000A79D 51                        <1>       push   ecx ; **
5030 0000A79E 53                        <1>       push   ebx ; ***
5031 0000A79F 57                        <1>       push   edi ; ****
5032                                    <1>
5033                                    <1>       ; 12/04/2016
5034 0000A7A0 8B5C240C                  <1>       mov    ebx, [esp+12]
5035                                    <1>
5036                                    <1> set_env_chk_validation2w:
5037 0000A7A4 89F7                      <1>       mov    edi, esi
5038 0000A7A6 4F                        <1>       dec    edi
5039                                    <1>
5040 0000A7A7 807FFF20                  <1>       cmp    byte [edi-1], 20h
5041 0000A7AB 771A                      <1>       ja     short set_env_chk_validation2z
5042                                    <1>
5043 0000A7AD 56                        <1>       push   esi
5044 0000A7AE 89FE                      <1>       mov    esi, edi
5045 0000A7B0 4E                        <1>       dec    esi
5046                                    <1>
5047                                    <1> set_env_chk_validation2x:
5048 0000A7B1 4E                        <1>       dec    esi
5049                                    <1>
5050 0000A7B2 39DE                      <1>       cmp    esi, ebx
5051 0000A7B4 7207                      <1>       jb     short set_env_chk_validation2y
5052                                    <1>
5053 0000A7B6 4F                        <1>       dec    edi
5054                                    <1>
5055 0000A7B7 8A06                      <1>       mov    al, [esi]
5056 0000A7B9 8807                      <1>       mov    [edi], al
5057                                    <1>
5058 0000A7BB EBF4                      <1>       jmp    short set_env_chk_validation2x
5059                                    <1>
5060                                    <1> set_env_chk_validation2y:
5061 0000A7BD 5E                        <1>       pop    esi
5062                                    <1>
5063                                    <1>       ;mov   byte [ebx], 20h
5064                                    <1>
5065 0000A7BE 43                        <1>       inc    ebx
5066 0000A7BF 895C240C                  <1>       mov    [esp+12], ebx
5067                                    <1>
5068 0000A7C3 FECC                      <1>       dec    ah ; 13/04/2016
5069                                    <1>
5070 0000A7C5 EBDD                      <1>       jmp    short set_env_chk_validation2w
5071                                    <1>
5072                                    <1> set_env_chk_validation2z:
5073 0000A7C7 BA00300900                <1>       mov    edx, Env_Page
5074 0000A7CC 89D7                      <1>       mov    edi, edx
5075                                    <1>
5076                                    <1> set_env_chk_validation3:
5077 0000A7CE AC                        <1>       lodsb
5078 0000A7CF 3C20                      <1>       cmp    al, 20h
5079 0000A7D1 74FB                      <1>       je     short set_env_chk_validation3
5080                                    <1>
5081 0000A7D3 9C                        <1>       pushf
5082                                    <1>
5083                                    <1>       ; 12/04/2016
5084                                    <1> set_env_chk_validation3n:
5085 0000A7D4 3C61                      <1>       cmp    al, 'a'
5086 0000A7D6 720C                      <1>       jb     short set_env_chk_validation3c
5087 0000A7D8 3C7A                      <1>       cmp    al, 'z'
5088 0000A7DA 7705                      <1>       ja     short set_env_chk_validation3x
5089 0000A7DC 2C20                      <1>       sub    al, 'a'-'A'
5090 0000A7DE 8846FF                    <1>       mov    [esi-1], al
5091                                    <1>
5092                                    <1> set_env_chk_validation3x:
5093 0000A7E1 AC                        <1>       lodsb
5094 0000A7E2 EBF0                      <1>       jmp    short set_env_chk_validation3n
5095                                    <1>
5096                                    <1> set_env_chk_validation3c:
5097 0000A7E4 3C20                      <1>       cmp    al, 20h
5098 0000A7E6 73F9                      <1>       jnb    short set_env_chk_validation3x
5099                                    <1>
5100 0000A7E8 803F00                    <1>       cmp    byte [edi], 0
```

```
5101 0000A7EB 7731              <1>       ja     short set_env_chk_validation4
5102                            <1>
5103 0000A7ED 9D                <1>       popf
5104 0000A7EE 7228              <1>       jb     short set_env_string_nothing
5105                            <1>
5106 0000A7F0 B900020000        <1>       mov    ecx, Env_Page_Size ; 512 (4096)
5107                            <1>
5108 0000A7F5 89DE              <1>       mov    esi, ebx ; 12/04/2016
5109                            <1>
5110                            <1> set_env_string_copy_to_envb:
5111 0000A7F7 AC                <1>       lodsb
5112 0000A7F8 3C20              <1>       cmp    al, 20h
5113 0000A7FA 720A              <1>       jb     short set_env_string_copy_to_envb_z
5114 0000A7FC AA                <1>       stosb
5115 0000A7FD E2F8              <1>       loop   set_env_string_copy_to_envb
5116                            <1>
5117                            <1>       ; 11/04/2016
5118 0000A7FF 89D7              <1>       mov    edi, edx ; Env_Page
5119 0000A801 B900020000        <1>       mov    ecx, Env_Page_Size
5120                            <1>
5121                            <1> set_env_string_copy_to_envb_z:
5122 0000A806 52                <1>       push   edx  ; Start address of the variable
5123 0000A807 BA00020000        <1>       mov    edx, Env_Page_Size
5124 0000A80C 29CA              <1>       sub    edx, ecx ; variable (string) length
5125                            <1>
5126 0000A80E 28C0              <1>       sub    al, al ; 0
5127 0000A810 F3AA              <1>       rep    stosb ; clear remain bytes of the env page
5128                            <1>
5129 0000A812 58                <1>       pop    eax  ; Start address of the variable
5130                            <1>
5131                            <1> set_env_string_allocate_envb_retn:  ; stc or clc return
5132 0000A813 5F                <1>       pop    edi ; ****
5133 0000A814 5B                <1>       pop    ebx ; ***
5134 0000A815 59                <1>       pop    ecx ; **
5135 0000A816 5E                <1>       pop    esi ; *
5136 0000A817 C3                <1>       retn
5137                            <1>
5138                            <1> set_env_string_nothing:
5139 0000A818 31C0              <1>       xor    eax, eax
5140 0000A81A 31D2              <1>       xor    edx, edx ; 11/04/2016
5141 0000A81C EBF5              <1>       jmp    short set_env_string_allocate_envb_retn
5142                            <1>
5143                            <1> set_env_chk_validation4:
5144                            <1>       ; 11/04/2016
5145 0000A81E 9D                <1>       popf
5146                            <1>
5147 0000A81F 89D6              <1>       mov    esi, edx  ; Env_Page
5148                            <1>
5149                            <1> set_env_chk_validation5:
5150 0000A821 89DF              <1>       mov    edi, ebx  ; ASCIIZ environment string address
5151 0000A823 0FB6CC            <1>       movzx  ecx, ah ; Variable (string) length (with '=')
5152                            <1>
5153                            <1> set_env_chk_validation5_loop:
5154 0000A826 AC                <1>       lodsb
5155 0000A827 AE                <1>       scasb
5156 0000A828 750A              <1>       jne    short set_env_chk_validation6
5157 0000A82A E2FA              <1>       loop   set_env_chk_validation5_loop
5158                            <1>
5159 0000A82C 3C3D              <1>       cmp    al, '='
5160 0000A82E 0F8483000000      <1>         je      set_env_change_variable
5161                            <1>
5162                            <1> set_env_chk_validation6:
5163 0000A834 08C0              <1>       or     al, al ; 0
5164 0000A836 7403              <1>       jz     short set_env_chk_validation7
5165                            <1>
5166 0000A838 AC                <1>       lodsb
5167 0000A839 EBF9              <1>       jmp    short set_env_chk_validation6
5168                            <1>
5169                            <1> set_env_chk_validation7:
5170 0000A83B 88E1              <1>       mov    cl, ah
5171 0000A83D 01F1              <1>       add    ecx, esi
5172 0000A83F 81F9FF310900      <1>       cmp    ecx, Env_Page + Env_Page_Size - 1
5173                            <1>              ; 511 (4095)
5174                            <1>              ; strlen + '=' + 0
5175 0000A845 72DA              <1>       jb     short set_env_chk_validation5
5176                            <1>
5177                            <1> set_env_chk_validation8: ; variable not found
5178 0000A847 0FB6F4            <1>       movzx  esi, ah  ; variable name length (with '=')
5179 0000A84A 01DE              <1>       add    esi, ebx ; position just after of the '='
5180                            <1>
5181                            <1> set_env_chk_validation8_loop:
5182 0000A84C AC                <1>       lodsb
5183 0000A84D 3C20              <1>       cmp    al, 20h
5184 0000A84F 74FB              <1>       je     short set_env_chk_validation8_loop
5185 0000A851 72C5              <1>       jb     short set_env_string_nothing
5186                            <1>
5187                            <1> set_env_chk_validation9:
5188 0000A853 AC                <1>       lodsb
5189 0000A854 3C20              <1>       cmp    al, 20h
5190 0000A856 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 0000A858 29DE              <1>       sub    esi, ebx ; variable+definition length
5196                            <1>
5197 0000A85A 56                <1>       push   esi ; *****
5198                            <1>
5199 0000A85B 89D6              <1>       mov    esi, edx ; Environment page address
5200                            <1>
5201 0000A85D B900020000        <1>       mov    ecx, Env_Page_Size ; 512 (4096)
5202                            <1>
5203                            <1> set_env_add_variable_loop:
5204 0000A862 AC                <1>       lodsb
5205 0000A863 20C0              <1>       and    al, al
```

```
5206 0000A865 7406              <1>       jz     short set_env_add_variable_chk1 ; 0
5207 0000A867 E2F9              <1>       loop   set_env_add_variable_loop
5208                            <1>
5209                            <1>       ; 11/04/2016
5210 0000A869 884EFF            <1>       mov    [esi-1], cl ; 0
5211 0000A86C 41                <1>       inc    ecx
5212                            <1>
5213                            <1> set_env_add_variable_chk1:
5214 0000A86D 49                <1>       dec    ecx
5215 0000A86E 7408              <1>       jz     short set_env_add_variable_nspc
5216 0000A870 AC                <1>       lodsb
5217 0000A871 08C0              <1>       or     al, al
5218 0000A873 740C              <1>       jz     short set_env_add_variable_chk2 ; 00
5219 0000A875 49                <1>       dec    ecx
5220 0000A876 75EA              <1>       jnz    short set_env_add_variable_loop
5221                            <1>
5222                            <1> set_env_add_variable_nspc: ; no space on environment page
5223 0000A878 58                <1>       pop    eax ; *****
5224 0000A879 B808000000        <1>       mov    eax, 8 ; No space for new environment string
5225 0000A87E F9                <1>       stc
5226 0000A87F EB92              <1>         jmp     short set_env_string_allocate_envb_retn
5227                            <1>
5228                            <1> set_env_add_variable_chk2:
5229 0000A881 8B0C24            <1>       mov    ecx, [esp] ; *****
5230 0000A884 4E                <1>       dec    esi ; beginning address of the new variable
5231 0000A885 89F0              <1>       mov    eax, esi
5232 0000A887 01C8              <1>       add    eax, ecx ; string length (with CR)
5233 0000A889 81C200020000      <1>       add    edx, Env_Page_Size ; 512 (4096)
5234 0000A88F 39D0              <1>       cmp    eax, edx
5235 0000A891 77E5              <1>       ja     short set_env_add_variable_nspc
5236 0000A893 49                <1>       dec    ecx ; except CR at the end
5237 0000A894 89CA              <1>       mov    edx, ecx ; 12/04/2016
5238 0000A896 89F7              <1>       mov    edi, esi
5239 0000A898 893C24            <1>       mov    [esp], edi ; ***** ; Start address of new variable
5240 0000A89B 89DE              <1>       mov    esi, ebx ; ASCIIZ environment string address
5241 0000A89D F3A4              <1>       rep    movsb
5242 0000A89F 28C0              <1>       sub    al, al
5243 0000A8A1 AA                <1>       stosb
5244 0000A8A2 58                <1>       pop    eax ; ***** ; Beginning address of new variable
5245 0000A8A3 81FF00320900      <1>         cmp     edi, Env_Page + Env_Page_Size ; 12/04/2016
5246 0000A8A9 0F8364FFFFFF      <1>         jnb     set_env_string_allocate_envb_retn ; OK !
5247 0000A8AF 880F              <1>       mov    [edi], cl ; 0
5248 0000A8B1 F8                <1>       clc    ; 13/04/2016
5249 0000A8B2 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 0000A8B7 8825[C08D0100]     <1>       mov    [env_var_length], ah
5258                            <1>
5259 0000A8BD 28C9              <1>       sub    cl, cl ; ecx = 0
5260                            <1>
5261 0000A8BF 57                <1>       push   edi ; *****
5262                            <1>
5263 0000A8C0 89F7              <1>       mov    edi, esi ; 11/04/2016
5264                            <1>
5265                            <1> set_env_change_variable_calc1:
5266 0000A8C2 AC                <1>       lodsb
5267 0000A8C3 08C0              <1>       or     al, al
5268 0000A8C5 7403              <1>       jz     short set_env_change_variable_calc2
5269                            <1>
5270 0000A8C7 41                <1>       inc    ecx ; length of environment string (after the '=')
5271                            <1>
5272 0000A8C8 EBF8              <1>       jmp    short set_env_change_variable_calc1
5273                            <1>
5274                            <1> set_env_change_variable_calc2:
5275 0000A8CA 8B3424            <1>       mov    esi, [esp] ; ASCIIZ environment string address
5276                            <1>
5277 0000A8CD 29D2              <1>       sub    edx, edx
5278                            <1>
5279                            <1> set_env_change_variable_calc3:
5280 0000A8CF AC                <1>       lodsb
5281 0000A8D0 3C20              <1>       cmp    al, 20h
5282 0000A8D2 7203              <1>       jb     short set_env_change_variable_calc4
5283                            <1>
5284 0000A8D4 42                <1>       inc    edx ; length of ASCIIZ string (after the '=')
5285                            <1>
5286 0000A8D5 EBF8              <1>       jmp    short set_env_change_variable_calc3
5287                            <1>
5288                            <1> set_env_change_variable_calc4:
5289 0000A8D7 C646FF00          <1>       mov    byte [esi-1], 0  ; put ZERO instead of CR
5290                            <1>
5291 0000A8DB 5E                <1>       pop    esi ; ***** ; ASCIIZ string address (after '=')
5292                            <1>
5293                            <1>       ; EDI = Old variable's address (after '=')
5294                            <1>
5295                            <1>       ; compare the new string with the old string
5296 0000A8DC 39CA              <1>       cmp    edx, ecx
5297 0000A8DE 7717              <1>       ja     short set_env_change_variable_calc5 ; longer
5298 0000A8E0 0F828F000000      <1>         jb      set_env_change_variable_calc9 ; shorter
5299                            <1>
5300                            <1>       ;same length (simple copy)
5301 0000A8E6 0FB6C4            <1>       movzx  eax, ah
5302 0000A8E9 01C2              <1>       add    edx, eax
5303 0000A8EB F7D8              <1>       neg    eax
5304 0000A8ED 01F8              <1>       add    eax, edi
5305                            <1>       ; EAX = Start address of the variable
5306                            <1>       ; EDX = Variable length (without ZERO at the end of variable)
5307                            <1>
5308 0000A8EF F3A4              <1>       rep    movsb
5309 0000A8F1 F8                <1>       clc    ; 13/04/2016
5310 0000A8F2 E91CFFFFFF        <1>         jmp     set_env_string_allocate_envb_retn ; OK !
```

```
                                <1>
                                <1> set_env_change_variable_calc5:
                                <1>       ; 11/04/2016
0000A8F7 52                     <1>       push   edx ; *****
0000A8F8 29CA                   <1>       sub    edx, ecx ; difference ; (the new string is longer)
0000A8FA 89F3                   <1>       mov    ebx, esi
0000A8FC 89FE                   <1>       mov    esi, edi
                                <1>
                                <1> set_env_change_variable_calc6:
0000A8FE AC                     <1>       lodsb
0000A8FF 20C0                   <1>       and    al, al
0000A901 75FB                   <1>       jnz    short set_env_change_variable_calc6
                                <1>
0000A903 81FE00320900           <1>       cmp    esi, Env_Page + Env_Page_Size ; 512 (4096)
0000A909 0F8369FFFFFF           <1>         jnb     set_env_add_variable_nspc
                                <1>
0000A90F 89F9                   <1>       mov    ecx, edi  ; current (old) variable's address
0000A911 89F7                   <1>       mov    edi, esi  ; next variable's address
                                <1>
0000A913 AC                     <1>       lodsb
0000A914 08C0                   <1>       or     al, al
0000A916 7416                   <1>       jz     short set_env_change_variable_calc8 ; 00
                                <1>
                                <1> set_env_change_variable_calc7:
0000A918 AC                     <1>       lodsb
0000A919 20C0                   <1>       and    al, al
0000A91B 75FB                   <1>       jnz    short set_env_change_variable_calc7
                                <1>
0000A91D 81FE00320900           <1>       cmp    esi, Env_Page + Env_Page_Size ; 512 (4096)
0000A923 0F834FFFFFFF           <1>         jnb     set_env_add_variable_nspc
                                <1>
0000A929 AC                     <1>       lodsb
0000A92A 08C0                   <1>       or     al, al
0000A92C 75EA                   <1>       jnz    short set_env_change_variable_calc7
                                <1>
                                <1> set_env_change_variable_calc8:
0000A92E 4E                     <1>       dec    esi ; address of the second (last) 0 of the 00
                                <1>
0000A92F 01F2                   <1>       add    edx, esi ; final position of the last 0
                                <1>
0000A931 81FA00320900           <1>       cmp    edx, Env_Page + Env_Page_Size ; 512 (4096)
0000A937 0F833BFFFFFF           <1>         jnb     set_env_add_variable_nspc
                                <1>
0000A93D 89C8                   <1>       mov    eax, ecx ; old variable's address (after '=')
                                <1>
0000A93F 89F1                   <1>       mov    ecx, esi
0000A941 29F9                   <1>       sub    ecx, edi ; count of bytes to move forward
                                <1>
                                <1>       ; 13/04/2016
0000A943 C60200                 <1>       mov    byte [edx], 0
0000A946 89D7                   <1>       mov    edi, edx
0000A948 29F2                   <1>       sub    edx, esi ; difference (additional byte count)
0000A94A 4F                     <1>       dec    edi ; the last zero address (first byte of the 00)
0000A94B 89FE                   <1>       mov    esi, edi
0000A94D 29D6                   <1>       sub    esi, edx ; - displacement
                                <1>
0000A94F FA                     <1>       cli    ; disable interrupts
0000A950 FD                     <1>       std    ; backward
                                <1>
0000A951 F3A4                   <1>       rep    movsb ; move ECX bytes from DS:ESI to ES:EDI
                                <1>
0000A953 FC                     <1>       cld    ; forward (default)
0000A954 FB                     <1>       sti    ; enable interrupts
                                <1>
0000A955 89C7                   <1>       mov    edi, eax
0000A957 59                     <1>       pop    ecx ; ***** ; byte count (after '=')
0000A958 89CA                   <1>       mov    edx, ecx
0000A95A 89DE                   <1>       mov    esi, ebx ; ASCIIZ string address (after '=')
0000A95C 89FB                   <1>       mov    ebx, edi
                                <1>
0000A95E F3A4                   <1>       rep    movsb
                                <1>
0000A960 880F                   <1>       mov    [edi], cl ; 0 ; end of variable
                                <1>
0000A962 0FB605[C08D0100]       <1>       movzx  eax, byte [env_var_length]
0000A969 01C2                   <1>       add    edx, eax ; variable length (total)
0000A96B F7D8                   <1>       neg    eax
0000A96D 01D8                   <1>       add    eax, ebx ; start address of the variable
0000A96F F8                     <1>       clc    ; 13/04/2016
0000A970 E99EFEFFFF             <1>         jmp     set_env_string_allocate_envb_retn ; OK !
                                <1>
                                <1> set_env_change_variable_calc9:
                                <1>       ; 11/04/2016
0000A975 21D2                   <1>       and    edx, edx ; is empty ?
0000A977 753B                   <1>       jnz    short set_env_change_variable_calc15
                                <1>
0000A979 0FB6DC                 <1>       movzx  ebx, ah
0000A97C F7DB                   <1>       neg    ebx
0000A97E 01FB                   <1>       add    ebx, edi
                                <1>
                                <1>       ; EBX = Start address of the variable (in env page)
                                <1>       ; EDX = Variable length = 0
                                <1>
0000A980 89FE                   <1>       mov    esi, edi
                                <1>
                                <1> set_env_change_variable_calc10:
0000A982 AC                     <1>       lodsb
0000A983 08C0                   <1>       or     al, al
0000A985 75FB                   <1>       jnz    short set_env_change_variable_calc10
                                <1>
0000A987 B9FF310900             <1>       mov    ecx, Env_Page + Env_Page_Size - 1
                                <1>
0000A98C 39CE                   <1>       cmp    esi, ecx ; +511 (+4095)
0000A98E 7604                   <1>       jna    short set_env_change_variable_calc11
                                <1>
```

```
5416 0000A990 89CE              <1>       mov    esi, ecx
5417 0000A992 8806              <1>       mov    [esi], al ; 0
5418                            <1>
5419                            <1> set_env_change_variable_calc11:
5420 0000A994 89DF              <1>       mov    edi, ebx ; old variable's start address
5421                            <1>
5422                            <1> set_env_change_variable_calc12:
5423 0000A996 AC                <1>       lodsb
5424 0000A997 AA                <1>       stosb
5425 0000A998 20C0              <1>       and    al, al
5426 0000A99A 75FA              <1>       jnz    short set_env_change_variable_calc12
5427 0000A99C 39CE              <1>       cmp    esi, ecx
5428 0000A99E 7706              <1>       ja     short set_env_change_variable_calc13
5429 0000A9A0 AC                <1>       lodsb
5430 0000A9A1 AA                <1>       stosb
5431 0000A9A2 20C0              <1>       and    al, al
5432 0000A9A4 75F0              <1>       jnz    short set_env_change_variable_calc12
5433                            <1>
5434                            <1> set_env_change_variable_calc13:
5435 0000A9A6 29F9              <1>       sub    ecx, edi
5436 0000A9A8 7203              <1>       jb     short set_env_change_variable_calc14
5437 0000A9AA 41                <1>       inc    ecx ; 1-512 (1-4096)
5438 0000A9AB F3AA              <1>       rep    stosb ; al = 0
5439                            <1>
5440                            <1> set_env_change_variable_calc14:
5441 0000A9AD 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 0000A9AF E95FFEFFFF        <1>         jmp     set_env_string_allocate_envb_retn ; OK !
5446                            <1>
5447                            <1> set_env_change_variable_calc15:
5448 0000A9B4 52                <1>       push   edx ; *****
5449 0000A9B5 F7DA              <1>       neg    edx
5450 0000A9B7 01CA              <1>       add    edx, ecx ; difference (the old string is longer)
5451 0000A9B9 89F3              <1>       mov    ebx, esi
5452 0000A9BB 89FE              <1>       mov    esi, edi
5453                            <1>
5454                            <1> set_env_change_variable_calc16:
5455 0000A9BD AC                <1>       lodsb
5456 0000A9BE 20C0              <1>       and    al, al
5457 0000A9C0 75FB              <1>       jnz    short set_env_change_variable_calc16
5458                            <1>
5459 0000A9C2 B900320900        <1>       mov    ecx, Env_Page + Env_Page_Size
5460                            <1>
5461 0000A9C7 39CE              <1>       cmp    esi, ecx ; +512 (+4096)
5462 0000A9C9 7605              <1>       jna    short set_env_change_variable_calc17
5463                            <1>
5464 0000A9CB 89CE              <1>       mov    esi, ecx
5465 0000A9CD 8846FF            <1>       mov    [esi-1], al ; 0
5466                            <1>
5467                            <1> set_env_change_variable_calc17:
5468 0000A9D0 89F9              <1>       mov    ecx, edi  ; current (old) variable's address
5469 0000A9D2 89F7              <1>       mov    edi, esi  ; next variable's address
5470                            <1>
5471 0000A9D4 AC                <1>       lodsb
5472 0000A9D5 08C0              <1>       or     al, al
5473 0000A9D7 741D              <1>       jz     short set_env_change_variable_calc20
5474                            <1>
5475                            <1> set_env_change_variable_calc18:
5476 0000A9D9 AC                <1>       lodsb
5477 0000A9DA 20C0              <1>       and    al, al
5478 0000A9DC 75FB              <1>       jnz    short set_env_change_variable_calc18
5479                            <1>
5480 0000A9DE 81FE00320900      <1>       cmp    esi, Env_Page + Env_Page_Size
5481 0000A9E4 720B              <1>       jb     short set_env_change_variable_calc19
5482 0000A9E6 740E              <1>       je     short set_env_change_variable_calc20
5483                            <1>
5484 0000A9E8 BEFF310900        <1>       mov    esi, Env_Page + Env_Page_Size - 1
5485 0000A9ED 8806              <1>       mov    [esi], al ; 0
5486 0000A9EF EB06              <1>       jmp    short set_env_change_variable_calc21
5487                            <1>
5488                            <1> set_env_change_variable_calc19:
5489 0000A9F1 AC                <1>       lodsb
5490 0000A9F2 08C0              <1>       or     al, al
5491 0000A9F4 75E3              <1>       jnz    short set_env_change_variable_calc18
5492                            <1>
5493                            <1> set_env_change_variable_calc20:
5494 0000A9F6 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 0000A9F7 89C8              <1>       mov    eax, ecx ; old variable's address (after '=')
5500                            <1>
5501 0000A9F9 89F1              <1>       mov    ecx, esi
5502 0000A9FB 29F9              <1>       sub    ecx, edi ; count of bytes to move backward
5503                            <1>
5504 0000A9FD 89FE              <1>       mov    esi, edi ; next variable's address
5505 0000A9FF 29D7              <1>       sub    edi, edx ; (displacement)
5506                            <1>
5507 0000AA01 F3A4              <1>       rep    movsb
5508                            <1>
5509 0000AA03 880F              <1>       mov    [edi], cl ; 0 ; 00 ; end of environment variables
5510                            <1>
5511 0000AA05 89C7              <1>       mov    edi, eax
5512 0000AA07 5A                <1>       pop    edx ; ***** ; byte count (after '=')
5513 0000AA08 89D1              <1>       mov    ecx, edx
5514 0000AA0A 89DE              <1>       mov    esi, ebx ; ASCIIZ string address (after '=')
5515 0000AA0C 89FB              <1>       mov    ebx, edi
5516                            <1>
5517 0000AA0E F3A4              <1>       rep    movsb
5518                            <1>
5519 0000AA10 880F              <1>       mov    [edi], cl ; 0 ; end of variable
5520                            <1>
```

```
5521 0000AA12 0FB605[C08D0100]    <1>       movzx  eax, byte [env_var_length]
5522 0000AA19 01C2                <1>       add    edx, eax ; variable length (total)
5523 0000AA1B F7D8                <1>       neg    eax
5524 0000AA1D 01D8                <1>       add    eax, ebx ; start address of the variable
5525 0000AA1F F8                  <1>       clc    ; 13/04/2016
5526 0000AA20 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 0000AA25 BE[69380100]        <1>       mov    esi, MainProgCfgFile
5536 0000AA2A 66B80018            <1>       mov    ax, 1800h ; Except volume label and dirs
5537 0000AA2E E83EEAFFFF          <1>       call   find_first_file
5538 0000AA33 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 0000AA35 A3[44820100]        <1>       mov    [MainProgCfg_FileSize], eax
5548                              <1>
5549 0000AA3A 668B5614            <1>       mov    dx, [esi+DirEntry_FstClusHI]
5550 0000AA3E C1E210              <1>       shl    edx, 16
5551 0000AA41 668B561A            <1>       mov    dx, [esi+DirEntry_FstClusLO]
5552 0000AA45 8915[748D0100]      <1>       mov    [csftdf_sf_cluster], edx
5553                              <1>
5554 0000AA4B 89C1                <1>       mov    ecx, eax
5555 0000AA4D 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 0000AA4F E81FBAFFFF          <1>       call   allocate_memory_block
5564 0000AA54 7235                <1>       jc     short loc_load_mainprog_cfg_exit
5565                              <1>
5566 0000AA56 A3[6C8D0100]        <1>       mov    [csftdf_sf_mem_addr], eax ; loading address
5567 0000AA5B 890D[708D0100]      <1>       mov    [csftdf_sf_mem_bsize], ecx ; block size
5568                              <1>
5569 0000AA61 31DB                <1>       xor    ebx, ebx
5570                              <1>       ;mov   [csftdf_sf_rbytes], ebx ; 0, reset
5571                              <1>
5572 0000AA63 8A3D[56820100]      <1>       mov    bh, [Current_Drv] ; [FindFile_Drv]
5573 0000AA69 BE00010900          <1>       mov    esi, Logical_DOSDisks
5574 0000AA6E 01DE                <1>       add    esi, ebx
5575                              <1>
5576 0000AA70 8B1D[6C8D0100]      <1>       mov    ebx, [csftdf_sf_mem_addr] ; memory block address
5577                              <1>
5578 0000AA76 807E0300            <1>       cmp    byte [esi+LD_FATType], 0
5579 0000AA7A 7710                <1>         ja   short loc_mcfg_load_fat_file
5580                              <1>
5581 0000AA7C C705[7C8D0100]0000- <1>       mov    dword [csftdf_r_size], 65536
5581 0000AA84 0100                <1>
5582 0000AA86 E9A1010000          <1>         jmp     loc_mcfg_load_fs_file
5583                              <1>
5584                              <1> loc_load_mainprog_cfg_exit:
5585 0000AA8B C3                  <1>       retn
5586                              <1>
5587                              <1> loc_mcfg_load_fat_file:
5588 0000AA8C 0FB74611            <1>       movzx  eax, word [esi+LD_BPB+BytesPerSec]
5589 0000AA90 0FB64E13            <1>       movzx  ecx, byte [esi+LD_BPB+SecPerClust]
5590 0000AA94 F7E1                <1>       mul    ecx
5591 0000AA96 A3[7C8D0100]        <1>       mov    [csftdf_r_size], eax
5592                              <1>
5593                              <1> loc_mcfg_load_fat_file_next:
5594 0000AA9B E822010000          <1>       call   mcfg_read_fat_file_sectors
5595 0000AAA0 0F8206010000        <1>         jc     mcfg_deallocate_mem
5596                              <1>
5597 0000AAA6 09D2                <1>       or     edx, edx ; edx > 0 -> EOF
5598 0000AAA8 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 0000AAAA C705[108E0100]-     <1>       mov    dword [mainprog_return_addr], loc_mcfg_ci_return_addr
5602 0000AAB0 [6DAB0000]          <1>
5603                              <1>       ;
5604 0000AAB4 8B35[6C8D0100]      <1>       mov    esi, [csftdf_sf_mem_addr]
5605 0000AABA 8935[48820100]      <1>       mov    [MainProgCfg_LineOffset], esi
5606                              <1>
5607 0000AAC0 A1[44820100]        <1>       mov    eax, [MainProgCfg_FileSize]
5608 0000AAC5 89C2                <1>       mov    edx, eax
5609 0000AAC7 01F2                <1>       add    edx, esi
5610                              <1>
5611                              <1> loc_mcfg_process_next_line_check:
5612 0000AAC9 89C1                <1>       mov    ecx, eax
5613                              <1>
5614 0000AACB 803E2A              <1>       cmp    byte [esi], "*" ; Remark sign
5615 0000AACE 7503                <1>       jne    short loc_mcfg_process_next_line
5616 0000AAD0 46                  <1>       inc    esi
5617 0000AAD1 EB17                <1>       jmp    short loc_move_mainprog_cfg_nl1
5618                              <1>
5619                              <1> loc_mcfg_process_next_line:
5620 0000AAD3 83F94F              <1>       cmp    ecx, 79
5621 0000AAD6 7605                <1>       jna    short loc_start_mainprog_cfg_process
5622                              <1>
5623 0000AAD8 B94F000000          <1>       mov    ecx, 79
```

```
                                <1>
                                <1> loc_start_mainprog_cfg_process:
0000AADD BF[06830100]           <1>       mov    edi, CommandBuffer
                                <1>
                                <1> loc_move_mainprog_cfg_line:
0000AAE2 AC                     <1>       lodsb
0000AAE3 3C20                   <1>       cmp    al, 20h
0000AAE5 720C                   <1>       jb     short loc_move_mainprog_cfg_nl2
0000AAE7 AA                     <1>       stosb
0000AAE8 E2F8                   <1>       loop   loc_move_mainprog_cfg_line
                                <1>
                                <1> loc_move_mainprog_cfg_nl1:
0000AAEA 39D6                   <1>       cmp    esi, edx ; + configuration file size
0000AAEC 7312                   <1>       jnb    short loc_end_of_mainprog_cfg_line
0000AAEE AC                     <1>       lodsb
0000AAEF 3C20                   <1>       cmp    al, 20h
0000AAF1 73F7                   <1>       jnb    short loc_move_mainprog_cfg_nl1
                                <1>
                                <1> loc_move_mainprog_cfg_nl2:
0000AAF3 39D6                   <1>       cmp    esi, edx
0000AAF5 7309                   <1>       jnb    short loc_end_of_mainprog_cfg_line
0000AAF7 8A06                   <1>       mov    al, [esi]
0000AAF9 3C20                   <1>       cmp    al, 20h
0000AAFB 7703                   <1>       ja     short loc_end_of_mainprog_cfg_line
0000AAFD 46                     <1>       inc    esi
0000AAFE EBF3                   <1>       jmp    short loc_move_mainprog_cfg_nl2
                                <1>
                                <1> loc_end_of_mainprog_cfg_line:
0000AB00 C60700                 <1>       mov    byte [edi], 0
                                <1>
0000AB03 8935[48820100]         <1>       mov    [MainProgCfg_LineOffset], esi
                                <1>
                                <1>       ; 22/11/2017
0000AB09 BE[0E830100]           <1>       mov    esi, CommandBuffer + 8
0000AB0E 29FE                   <1>       sub    esi, edi
0000AB10 7606                   <1>       jna    short loc_move_mainprog_cfg_command
0000AB12 30C0                   <1>       xor    al, al
                                <1> loc_mainprog_cfg_clear_chrs:
0000AB14 AA                     <1>       stosb
0000AB15 4E                     <1>       dec    esi
0000AB16 75FC                   <1>       jnz    short loc_mainprog_cfg_clear_chrs
                                <1>
                                <1> loc_move_mainprog_cfg_command:
0000AB18 BE[06830100]           <1>       mov    esi, CommandBuffer
0000AB1D 89F7                   <1>       mov    edi, esi
0000AB1F 31DB                   <1>       xor    ebx, ebx
                                <1>       ;xor   ecx, ecx
0000AB21 30C9                   <1>       xor    cl, cl
                                <1>
                                <1> loc_move_mcfg_first_cmd_char:
0000AB23 8A041E                 <1>       mov    al, [esi+ebx]
0000AB26 FEC3                   <1>       inc    bl
0000AB28 3C20                   <1>       cmp    al, 20h
0000AB2A 7712                   <1>       ja     short loc_move_mcfg_cmd_capitalizing
0000AB2C 7237                   <1>       jb     short loc_move_mcfg_cmd_arguments_ok
0000AB2E 80FB4F                 <1>       cmp    bl, 79
0000AB31 72F0                   <1>       jb     short loc_move_mcfg_first_cmd_char
0000AB33 EB30                   <1>       jmp    short loc_move_mcfg_cmd_arguments_ok
                                <1>
                                <1> loc_move_mcfg_next_cmd_char:
0000AB35 8A041E                 <1>       mov    al, [esi+ebx]
0000AB38 FEC3                   <1>       inc    bl
0000AB3A 3C20                   <1>       cmp    al, 20h
0000AB3C 7614                   <1>       jna    short loc_move_mcfg_cmd_ok
                                <1>
                                <1> loc_move_mcfg_cmd_capitalizing:
0000AB3E 3C61                   <1>       cmp    al, 61h ; 'a'
0000AB40 7206                   <1>       jb     short loc_move_mcfg_cmd_caps_ok
0000AB42 3C7A                   <1>       cmp    al, 7Ah ; 'z'
0000AB44 7702                   <1>       ja     short loc_move_mcfg_cmd_caps_ok
0000AB46 24DF                   <1>       and    al, 0DFh ; sub      al, 'a'-'A'
                                <1>
                                <1> loc_move_mcfg_cmd_caps_ok:
0000AB48 AA                     <1>       stosb
0000AB49 FEC1                   <1>       inc    cl
0000AB4B 80FB4F                 <1>       cmp    bl, 79
0000AB4E 72E5                   <1>       jb     short loc_move_mcfg_next_cmd_char
0000AB50 EB13                   <1>       jmp    short loc_move_mcfg_cmd_arguments_ok
                                <1>
                                <1> loc_move_mcfg_cmd_ok:
0000AB52 30C0                   <1>       xor    al, al ; 0
                                <1>
                                <1> loc_move_mcfg_cmd_arguments:
0000AB54 8807                   <1>       mov    [edi], al
0000AB56 47                     <1>       inc    edi
0000AB57 80FB4F                 <1>       cmp    bl, 79
0000AB5A 7309                   <1>       jnb    short loc_move_mcfg_cmd_arguments_ok
0000AB5C 8A041E                 <1>       mov    al, [esi+ebx]
0000AB5F FEC3                   <1>       inc    bl
0000AB61 3C20                   <1>       cmp    al, 20h
0000AB63 73EF                   <1>       jnb    short loc_move_mcfg_cmd_arguments
                                <1>
                                <1> loc_move_mcfg_cmd_arguments_ok:
0000AB65 C60700                 <1>       mov    byte [edi], 0
                                <1>
                                <1> loc_mcfg_process_cmd_interpreter:
0000AB68 E825E0FFFF             <1>       call    command_interpreter
                                <1>
                                <1> loc_mcfg_ci_return_addr:
0000AB6D A1[44820100]           <1>       mov    eax, [MainProgCfg_FileSize]
0000AB72 89C2                   <1>       mov    edx, eax
0000AB74 8B35[48820100]         <1>       mov    esi, [MainProgCfg_LineOffset]
0000AB7A 01F2                   <1>       add    edx, esi
0000AB7C 0305[6C8D0100]         <1>       add    eax, [csftdf_sf_mem_addr]
0000AB82 29F0                   <1>       sub    eax, esi
```

```
5729 0000AB84 0F873FFFFFFF        <1>         ja      loc_mcfg_process_next_line_check
5730                              <1>
5731 0000AB8A E81D000000          <1>       call   mcfg_deallocate_mem
5732                              <1>
5733 0000AB8F B94F000000          <1>       mov    ecx, 79 ; 80 ?
5734 0000AB94 BF[06830100]        <1>       mov    edi, CommandBuffer
5735 0000AB99 30C0                <1>       xor    al, al
5736 0000AB9B F3AA                <1>       rep    stosb
5737                              <1>
5738                              <1>       ; 06/05/2016
5739 0000AB9D BE[BF440100]        <1>       mov    esi, nextline
5740 0000ABA2 E89EC9FFFF          <1>       call   print_msg
5741 0000ABA7 E963D6FFFF          <1>       jmp    dos_prompt
5742                              <1>
5743                              <1> mcfg_deallocate_mem:
5744 0000ABAC A1[6C8D0100]        <1>       mov    eax, [csftdf_sf_mem_addr] ; start address
5745 0000ABB1 8B0D[708D0100]      <1>       mov    ecx, [csftdf_sf_mem_bsize] ; block size
5746                              <1>       ;call  deallocate_memory_block
5747                              <1>       ;retn
5748 0000ABB7 E9C4BAFFFF          <1>       jmp    deallocate_memory_block
5749                              <1>
5750                              <1> mcfg_read_file_sectors:
5751                              <1>       ; 14/04/2016
5752 0000ABBC 807E0300            <1>       cmp    byte [esi+LD_FATType], 0
5753 0000ABC0 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 0000ABC2 8B15[44820100]      <1>       mov    edx, [MainProgCfg_FileSize]
5763 0000ABC8 2B15[848D0100]      <1>       sub    edx, [csftdf_sf_rbytes]
5764 0000ABCE 3B15[7C8D0100]      <1>       cmp    edx, [csftdf_r_size]
5765 0000ABD4 7306                <1>       jnb    short mcfg_read_fat_file_secs_1
5766 0000ABD6 8915[7C8D0100]      <1>       mov    [csftdf_r_size], edx
5767                              <1>
5768                              <1> mcfg_read_fat_file_secs_1:
5769 0000ABDC A1[7C8D0100]        <1>       mov    eax, [csftdf_r_size]
5770 0000ABE1 29D2                <1>       sub    edx, edx
5771 0000ABE3 0FB74E11            <1>       movzx  ecx, word [esi+LD_BPB+BytesPerSec]
5772 0000ABE7 01C8                <1>       add    eax, ecx
5773 0000ABE9 48                  <1>       dec    eax
5774 0000ABEA F7F1                <1>       div    ecx
5775 0000ABEC 89C1                <1>       mov    ecx, eax ; sector count
5776 0000ABEE A1[748D0100]        <1>       mov    eax, [csftdf_sf_cluster]
5777                              <1>
5778                              <1>       ; EBX = memory block address (current)
5779                              <1>
5780 0000ABF3 E88C230000          <1>       call   read_fat_file_sectors
5781 0000ABF8 7230                <1>       jc     short mcfg_read_fat_file_secs_3
5782                              <1>
5783                              <1>       ; EBX = next memory address
5784                              <1>
5785 0000ABFA A1[848D0100]        <1>       mov    eax, [csftdf_sf_rbytes]
5786 0000ABFF 0305[7C8D0100]      <1>       add    eax, [csftdf_r_size]
5787 0000AC05 8B15[44820100]      <1>       mov    edx, [MainProgCfg_FileSize]
5788 0000AC0B 39D0                <1>       cmp    eax, edx
5789 0000AC0D 731B                <1>       jnb    short mcfg_read_fat_file_secs_3 ; edx > 0
5790 0000AC0F A3[848D0100]        <1>       mov    [csftdf_sf_rbytes], eax
5791                              <1>
5792 0000AC14 53                  <1>       push   ebx ; *
5793                              <1>       ; get next cluster (csftdf_r_size! bytes)
5794 0000AC15 A1[748D0100]        <1>       mov    eax, [csftdf_sf_cluster]
5795 0000AC1A E837210000          <1>       call   get_next_cluster
5796 0000AC1F 5B                  <1>       pop    ebx ; *
5797 0000AC20 7301                <1>       jnc    short mcfg_read_fat_file_secs_2
5798                              <1>
5799                              <1>       ;mov   eax, 17; Read error !
5800 0000AC22 C3                  <1>       retn
5801                              <1>
5802                              <1> mcfg_read_fat_file_secs_2:
5803 0000AC23 29D2                <1>       sub    edx, edx ; 0
5804 0000AC25 A3[748D0100]        <1>       mov    [csftdf_sf_cluster], eax ; next cluster
5805                              <1>
5806                              <1> mcfg_read_fat_file_secs_3:
5807 0000AC2A C3                  <1>       retn
5808                              <1>
5809                              <1> mcfg_read_fs_file_sectors:
5810 0000AC2B C3                  <1>       retn
5811                              <1>
5812                              <1> loc_mcfg_load_fs_file:
5813 0000AC2C C3                  <1>       retn
5814                              <1>
5815                              <1> load_and_execute_file:
5816                              <1>       ; 04/01/2017
5817                              <1>       ; 06/05/2016, 07/05/2016, 11/05/2016
5818                              <1>       ; 23/04/2016, 24/04/2016
5819                              <1>       ; 22/04/2016 (TRDOS 386 = TRDOS v2.0)
5820                              <1>       ; 05/11/2011
5821                              <1>       ; (TRDOS v1, CMDINTR.ASM, 'cmp_cmd_run', 'cmp_cmd_external')
5822                              <1>       ; ('loc_run_check_filename')
5823                              <1>       ; 29/08/2011
5824                              <1>       ; 10/09/2011
5825                              <1>       ; INPUT->
5826                              <1>       ;     ESI = Path Name address (CommandBuffer address)
5827                              <1>       ; OUTPUT ->
5828                              <1>       ;     none (error message will be shown if an error will occur)
5829                              <1>       ;
5830                              <1>       ; (EAX, EBX, ECX, EDX, ESI, EDI, EBP will be changed)
5831                              <1>       ;
5832                              <1> loc_run_check_filename:
5833 0000AC2D 803E20              <1>       cmp    byte [esi], 20h
```

```
5834 0000AC30 0F822BE3FFFF        <1>       jb    loc_cmd_failed
5835 0000AC36 7703                <1>       ja    short loc_run_check_filename_ok
5836 0000AC38 46                  <1>       inc   esi
5837 0000AC39 EBF2                <1>       jmp   short loc_run_check_filename
5838                              <1>
5839                              <1> loc_run_check_filename_ok:
5840 0000AC3B C605[B7820100]00    <1>       mov   byte [CmdArgStart], 0 ; reset
5841 0000AC42 56                  <1>       push  esi ; *
5842                              <1> loc_run_get_first_arg_pos:
5843 0000AC43 46                  <1>       inc   esi
5844 0000AC44 8A06                <1>       mov   al, [esi]
5845 0000AC46 3C20                <1>       cmp   al, 20h
5846 0000AC48 77F9                <1>       ja    short loc_run_get_first_arg_pos
5847 0000AC4A C60600              <1>       mov   byte [esi], 0
5848                              <1> loc_run_get_external_arg_pos:
5849                              <1>       ; 11/05/2016
5850 0000AC4D 46                  <1>       inc   esi
5851 0000AC4E 8A06                <1>       mov   al, [esi]
5852 0000AC50 3C20                <1>       cmp   al, 20h
5853 0000AC52 760C                <1>       jna   short loc_run_parse_path_name
5854 0000AC54 89F0                <1>       mov   eax, esi
5855 0000AC56 2D[06830100]        <1>       sub   eax, CommandBuffer
5856 0000AC5B A2[B7820100]        <1>       mov   byte [CmdArgStart], al
5857                              <1> loc_run_parse_path_name:
5858 0000AC60 5E                  <1>       pop   esi ; *
5859 0000AC61 BF[F68A0100]        <1>       mov   edi, FindFile_Drv
5860 0000AC66 E8D7090000          <1>       call  parse_path_name
5861 0000AC6B 0F82F0E2FFFF        <1>       jc    loc_cmd_failed
5862                              <1>
5863                              <1> loc_run_check_filename_exists:
5864 0000AC71 BE[388B0100]        <1>       mov   esi, FindFile_Name
5865 0000AC76 803E20              <1>       cmp   byte [esi], 20h
5866 0000AC79 0F86E2E2FFFF        <1>       jna   loc_cmd_failed
5867                              <1>
5868                              <1> loc_run_check_exe_filename_ext:
5869 0000AC7F E890020000          <1>       call  check_prg_filename_ext
5870 0000AC84 0F82D7E2FFFF        <1>       jc    loc_cmd_failed
5871                              <1>
5872                              <1> loc_run_check_exe_filename_ext_ok:
5873 0000AC8A 66A3[0E8E0100]      <1>       mov   word [EXE_ID], ax
5874                              <1>
5875                              <1> loc_run_drv:
5876 0000AC90 C605[0D8E0100]00    <1>       mov   byte [Run_Manual_Path], 0
5877 0000AC97 A1[50820100]        <1>       mov   eax, [Current_Dir_FCluster]
5878 0000AC9C A3[088E0100]        <1>         mov     [Run_CDirFC], eax
5879                              <1>       ;
5880 0000ACA1 8A35[56820100]      <1>       mov   dh, [Current_Drv]
5881 0000ACA7 8835[B2890100]      <1>       mov   [RUN_CDRV], dh
5882                              <1>
5883 0000ACAD 8A15[F68A0100]      <1>       mov   dl, [FindFile_Drv]
5884 0000ACB3 38F2                <1>       cmp   dl, dh
5885 0000ACB5 7412                <1>       je    short loc_run_change_directory
5886                              <1>
5887 0000ACB7 8005[0D8E0100]02    <1>       add   byte [Run_Manual_Path], 2
5888                              <1>
5889 0000ACBE E80BD4FFFF          <1>       call  change_current_drive
5890 0000ACC3 0F82C3E2FFFF        <1>       jc    loc_run_cmd_failed
5891                              <1>
5892                              <1> loc_run_change_directory:
5893 0000ACC9 803D[F78A0100]20    <1>       cmp   byte [FindFile_Directory], 20h
5894 0000ACD0 7623                <1>       jna   short loc_run_find_executable_file
5895                              <1>
5896 0000ACD2 FE05[0D8E0100]      <1>       inc   byte [Run_Manual_Path]
5897                              <1>
5898 0000ACD8 FE05[23380100]      <1>       inc   byte [Restore_CDIR]
5899                              <1>
5900 0000ACDE BE[F78A0100]        <1>       mov   esi, FindFile_Directory
5901 0000ACE3 30E4                <1>       xor   ah, ah ; CD_COMMAND sign -> 0
5902 0000ACE5 E842030000          <1>       call  change_current_directory
5903 0000ACEA 0F829CE2FFFF        <1>       jc    loc_run_cmd_failed
5904                              <1>
5905                              <1> loc_run_change_prompt_dir_string:
5906 0000ACF0 E857020000          <1>       call  change_prompt_dir_string
5907                              <1>
5908                              <1> loc_run_find_executable_file:
5909 0000ACF5 66C705[0C8E0100]00- <1>       mov   word [Run_Auto_Path], 0
5909 0000ACFD 00                  <1>
5910                              <1>
5911                              <1> loc_run_find_executable_file_next:
5912 0000ACFE BE[388B0100]        <1>       mov   esi, FindFile_Name
5913                              <1> loc_run_find_program_file_next:
5914 0000AD03 66B80018            <1>       mov   ax, 1800h ; Except volume label and dirs
5915 0000AD07 E865E7FFFF          <1>       call  find_first_file
5916                              <1>       ; ESI = Directory Entry (FindFile_DirEntry) Location
5917                              <1>       ; EDI = Directory Buffer Directory Entry Location
5918                              <1>       ; EAX = File size
5919 0000AD0C 0F835C010000        <1>       jnc   loc_load_and_run_file
5920                              <1>
5921 0000AD12 3C02                <1>       cmp   al, 2 ; file not found
5922 0000AD14 0F8572E2FFFF        <1>       jne   loc_run_cmd_failed
5923                              <1>
5924 0000AD1A 66A1[0E8E0100]      <1>       mov   ax, word [EXE_ID]
5925 0000AD20 80FC2E              <1>       cmp   ah, '.' ; File name has extension sign
5926 0000AD23 7424                <1>       je    short loc_run_check_auto_path
5927                              <1>
5928 0000AD25 08C0                <1>       or    al, al
5929 0000AD27 7520                <1>       jnz   short loc_run_check_auto_path
5930                              <1>
5931 0000AD29 80FC08              <1>       cmp   ah, 8 ; count of file name chars
5932 0000AD2C 771B                <1>       ja    short loc_run_check_auto_path
5933                              <1>
5934                              <1> loc_run_change_file_ext_to_prg:
5935 0000AD2E 0FB6DC              <1>       movzx ebx, ah ; count of file name chars
5936 0000AD31 BE[388B0100]        <1>       mov   esi, FindFile_Name
5937 0000AD36 01F3                <1>       add   ebx, esi
```

```
5938                                    <1>       ; 07/05/2016
5939 0000AD38 C7032E505247              <1>       mov    dword [ebx],  '.PRG'
5940 0000AD3E 66C705[0E8E0100]50-       <1>       mov    word [EXE_ID], 'P.'
5940 0000AD46 2E                        <1>
5941 0000AD47 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 0000AD49 A0[0D8E0100]              <1>       mov    al, [Run_Manual_Path]
5952 0000AD4E 08C0                      <1>       or     al, al
5953 0000AD50 0F850BE2FFFF              <1>       jnz    loc_cmd_failed
5954                                    <1>
5955                                    <1> loc_run_check_auto_path_again:
5956 0000AD56 66833D[0C8E0100]FF        <1>       cmp    word [Run_Auto_Path], 0FFFFh
5957                                    <1>              ; 0FFFFh = Not a valid run path (in ENV block)
5958 0000AD5E 0F83FDE1FFFF              <1>       jnb    loc_cmd_failed
5959                                    <1>       ; xor  al, al
5960 0000AD64 BE[EF380100]              <1>       mov    esi, Cmd_Path ; 'PATH'
5961 0000AD69 BF[56830100]              <1>       mov    edi, TextBuffer
5962 0000AD6E E848F9FFFF                <1>       call   get_environment_string
5963 0000AD73 730E                      <1>       jnc    short loc_run_chk_filename_ext_again
5964 0000AD75 66C705[0C8E0100]FF-       <1>       mov    word [Run_Auto_Path], 0FFFFh ; invalid
5964 0000AD7D FF                        <1>
5965 0000AD7E E9DEE1FFFF                <1>       jmp    loc_cmd_failed
5966                                    <1>
5967                                    <1> loc_run_chk_filename_ext_again:
5968 0000AD83 89C1                      <1>       mov    ecx, eax ; string length (with zero tail)
5969 0000AD85 49                        <1>       dec    ecx ; without zero tail
5970 0000AD86 66A1[0E8E0100]            <1>       mov    ax, [EXE_ID]
5971 0000AD8C 80FC2E                    <1>       cmp    ah, '.'
5972 0000AD8F 740E                      <1>       je     short loc_run_chk_auto_path_pos
5973                                    <1>
5974                                    <1> loc_run_change_file_ext_to_noext_again:
5975 0000AD91 0FB6DC                    <1>       movzx  ebx, ah
5976 0000AD94 BE[388B0100]              <1>       mov    esi, FindFile_Name
5977 0000AD99 01F3                      <1>       add    ebx, esi
5978 0000AD9B 29C0                      <1>       sub    eax, eax
5979 0000AD9D 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 0000AD9F 66A1[0C8E0100]            <1>       mov    ax, [Run_Auto_Path]
5984 0000ADA5 39C8                      <1>       cmp    eax, ecx ; ecx = string length (except zero tail)
5985 0000ADA7 0F83B4E1FFFF              <1>       jnb    loc_cmd_failed
5986                                    <1>       ;or    eax, eax
5987 0000ADAD 6609C0                    <1>       or     ax, ax
5988 0000ADB0 7502                      <1>       jnz    short loc_run_auto_path_pos_move
5989 0000ADB2 B005                      <1>       mov    al, 5
5990                                    <1>
5991                                    <1> loc_run_auto_path_pos_move:
5992 0000ADB4 89FE                      <1>       mov    esi, edi ; offset TextBuffer
5993 0000ADB6 01C6                      <1>       add    esi, eax
5994                                    <1>
5995                                    <1> loc_run_auto_path_pos_space_loop:
5996 0000ADB8 AC                        <1>       lodsb
5997 0000ADB9 3C20                      <1>       cmp    al, 20h
5998 0000ADBB 74FB                      <1>       je     short loc_run_auto_path_pos_space_loop
5999 0000ADBD 0F829EE1FFFF              <1>       jb     loc_cmd_failed
6000 0000ADC3 AA                        <1>       stosb
6001                                    <1> loc_run_auto_path_pos_move_next:
6002 0000ADC4 AC                        <1>       lodsb
6003 0000ADC5 3C3B                      <1>       cmp    al, ';'
6004 0000ADC7 7414                      <1>       je     short loc_run_auto_path_pos_move_last_byte
6005 0000ADC9 3C20                      <1>       cmp    al, 20h
6006 0000ADCB 74F7                      <1>       je     short loc_run_auto_path_pos_move_next
6007 0000ADCD 7203                      <1>       jb     short loc_byte_ptr_end_of_path
6008 0000ADCF AA                        <1>       stosb
6009 0000ADD0 EBF2                      <1>       jmp    short loc_run_auto_path_pos_move_next
6010                                    <1>
6011                                    <1> loc_byte_ptr_end_of_path:
6012 0000ADD2 66C705[0C8E0100]FF-       <1>       mov    word [Run_Auto_Path], 0FFFFh ; end of path
6012 0000ADDA FF                        <1>
6013 0000ADDB EB0D                      <1>       jmp    short loc_run_auto_path_move_ok
6014                                    <1>
6015                                    <1> loc_run_auto_path_pos_move_last_byte:
6016 0000ADDD 89F0                      <1>       mov    eax, esi
6017 0000ADDF 2D[56830100]              <1>       sub    eax, TextBuffer
6018 0000ADE4 66A3[0C8E0100]            <1>       mov    [Run_Auto_Path], ax ; next path position
6019                                    <1>
6020                                    <1> loc_run_auto_path_move_ok:
6021 0000ADEA 4F                        <1>       dec    edi
6022 0000ADEB B02F                      <1>       mov    al, '/'
6023 0000ADED 3807                      <1>       cmp    [edi], al
6024 0000ADEF 7403                      <1>       je     short loc_run_auto_path_move_file_name
6025 0000ADF1 47                        <1>       inc    edi
6026 0000ADF2 8807                      <1>       mov    [edi], al
6027                                    <1>
6028                                    <1> loc_run_auto_path_move_file_name:
6029 0000ADF4 47                        <1>       inc    edi
6030 0000ADF5 BE[388B0100]              <1>       mov    esi, FindFile_Name
6031                                    <1>
6032                                    <1> loc_run_auto_path_move_fn_loop:
6033 0000ADFA AC                        <1>       lodsb
6034 0000ADFB AA                        <1>       stosb
6035 0000ADFC 08C0                      <1>       or     al, al
6036 0000ADFE 75FA                      <1>       jnz    short loc_run_auto_path_move_fn_loop
6037                                    <1>
6038 0000AE00 BE[56830100]              <1>       mov    esi, TextBuffer
6039 0000AE05 BF[F68A0100]              <1>       mov    edi, FindFile_Drv
```

```
6040 0000AE0A E833080000          <1>       call   parse_path_name
6041 0000AE0F 0F824CE1FFFF        <1>       jc     loc_cmd_failed
6042                              <1>
6043 0000AE15 8A35[56820100]      <1>       mov    dh, [Current_Drv]
6044 0000AE1B 8A15[F68A0100]      <1>       mov    dl, [FindFile_Drv]
6045 0000AE21 38F2                <1>       cmp    dl, dh
6046 0000AE23 740B                <1>       je     short loc_run_change_directory_again
6047                              <1>
6048 0000AE25 E8A4D2FFFF          <1>       call   change_current_drive
6049 0000AE2A 0F825CE1FFFF        <1>       jc     loc_run_cmd_failed
6050                              <1>
6051                              <1> loc_run_change_directory_again:
6052 0000AE30 803D[F78A0100]20    <1>       cmp    byte [FindFile_Directory], 20h
6053 0000AE37 761D                <1>       jna    short loc_load_executable_cdir_chk_again
6054                              <1>
6055 0000AE39 FE05[23380100]      <1>       inc    byte [Restore_CDIR]
6056 0000AE3F BE[F78A0100]        <1>       mov    esi, FindFile_Directory
6057 0000AE44 30E4                <1>       xor    ah, ah ; CD_COMMAND sign -> 0
6058 0000AE46 E8E1010000          <1>       call   change_current_directory
6059 0000AE4B 0F823BE1FFFF        <1>       jc     loc_run_cmd_failed
6060                              <1>
6061                              <1> loc_run_chg_prompt_dir_str_again:
6062 0000AE51 E8F6000000          <1>       call   change_prompt_dir_string
6063                              <1>
6064                              <1> loc_load_executable_cdir_chk_again:
6065 0000AE56 A1[50820100]        <1>       mov    eax, [Current_Dir_FCluster]
6066 0000AE5B 3B05[088E0100]      <1>       cmp    eax, [Run_CDirFC]
6067 0000AE61 0F8597FEFFFF        <1>       jne    loc_run_find_executable_file_next
6068 0000AE67 30C0                <1>       xor    al, al ; 0
6069 0000AE69 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 0000AE6E BE[388B0100]        <1>       mov    esi, FindFile_Name
6076 0000AE73 BF[56830100]        <1>       mov    edi, TextBuffer
6077                              <1>
6078                              <1>       ; 24/04/2016
6079 0000AE78 31D2                <1>       xor    edx, edx
6080 0000AE7A 668915[4A040300]    <1>       mov    word [argc], dx ; 0
6081 0000AE81 8915[8C030300]      <1>       mov    dword [u.nread], edx ; 0
6082                              <1>
6083                              <1> loc_load_and_run_file_1:
6084 0000AE87 AC                  <1>       lodsb
6085 0000AE88 AA                  <1>       stosb
6086 0000AE89 FF05[8C030300]      <1>       inc    dword [u.nread]
6087 0000AE8F 20C0                <1>       and    al, al
6088 0000AE91 75F4                <1>       jnz    short loc_load_and_run_file_1
6089                              <1>
6090 0000AE93 A0[B7820100]        <1>       mov    al, [CmdArgStart]
6091 0000AE98 20C0                <1>       and    al, al
6092 0000AE9A 7445                <1>       jz     short loc_load_and_run_file_7
6093                              <1>
6094 0000AE9C 0FB6F0              <1>       movzx  esi, al ; 11/05/2016
6095 0000AE9F B950000000          <1>       mov    ecx, 80
6096 0000AEA4 29F1                <1>       sub    ecx, esi
6097 0000AEA6 81C6[06830100]      <1>       add    esi, CommandBuffer
6098                              <1>
6099 0000AEAC 66FF05[4A040300]    <1>       inc    word [argc] ; 11/05/2016
6100                              <1>
6101                              <1> loc_load_and_run_file_2:
6102 0000AEB3 AC                  <1>       lodsb
6103 0000AEB4 3C20                <1>       cmp    al, 20h
6104 0000AEB6 7717                <1>       ja     short loc_load_and_run_file_5
6105 0000AEB8 721E                <1>       jb     short loc_load_and_run_file_6
6106                              <1>
6107                              <1> loc_load_and_run_file_3:
6108 0000AEBA 803E20              <1>       cmp    byte [esi], 20h
6109 0000AEBD 7707                <1>       ja     short loc_load_and_run_file_4
6110 0000AEBF 7217                <1>       jb     short loc_load_and_run_file_6
6111 0000AEC1 46                  <1>       inc    esi
6112 0000AEC2 E2F6                <1>       loop   loc_load_and_run_file_3
6113 0000AEC4 EB12                <1>       jmp    short loc_load_and_run_file_6
6114                              <1>
6115                              <1> loc_load_and_run_file_4:
6116 0000AEC6 28C0                <1>       sub    al, al ; 0
6117 0000AEC8 66FF05[4A040300]    <1>       inc    word [argc]
6118                              <1> loc_load_and_run_file_5:
6119 0000AECF AA                  <1>       stosb
6120 0000AED0 FF05[8C030300]      <1>       inc    dword [u.nread]
6121 0000AED6 E2DB                <1>       loop   loc_load_and_run_file_2
6122                              <1>
6123                              <1> loc_load_and_run_file_6:
6124 0000AED8 30C0                <1>       xor    al, al ; 0
6125 0000AEDA AA                  <1>       stosb
6126 0000AEDB FF05[8C030300]      <1>       inc    dword [u.nread]
6127                              <1> loc_load_and_run_file_7:
6128 0000AEE1 8807                <1>       mov    [edi], al ; 0
6129 0000AEE3 66FF05[4A040300]    <1>       inc    word [argc] ; 24/04/2016
6130 0000AEEA FF05[8C030300]      <1>       inc    dword [u.nread] ; 24/04/2016
6131 0000AEF0 BE[56830100]        <1>       mov    esi, TextBuffer
6132 0000AEF5 8B15[648B0100]      <1>       mov    edx, [FindFile_DirEntry+DirEntry_FileSize]
6133 0000AEFB 66A1[5C8B0100]      <1>       mov    ax, [FindFile_DirEntry+DirEntry_FstClusHI]
6134 0000AF01 C1E010              <1>       shl    eax, 16 ; 13/11/2017
6135 0000AF04 66A1[628B0100]      <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 0000AF0A E8DA5B0000          <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 0000AF0F 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 0000AF14 30E4                     <1>         xor    ah, ah
6164                                    <1> loc_run_check_filename_ext:
6165 0000AF16 AC                       <1>         lodsb
6166 0000AF17 3C21                     <1>         cmp    al, 21h
6167 0000AF19 7229                     <1>         jb     short loc_check_exe_fn_retn
6168 0000AF1B FEC4                     <1>         inc    ah
6169 0000AF1D 3C2E                     <1>         cmp    al, '.'
6170 0000AF1F 75F5                     <1>         jne    short loc_run_check_filename_ext
6171                                    <1>
6172                                    <1> loc_run_check_filename_ext_dot:
6173 0000AF21 80FC02                   <1>         cmp    ah, 2 ; .??? is not valid
6174 0000AF24 88C4                     <1>         mov    ah, al ; '.'
6175 0000AF26 7219                     <1>         jb     short loc_check_prg_fn_retn
6176                                    <1>
6177                                    <1> loc_run_check_filename_ext_dot_ok:
6178 0000AF28 AC                       <1>         lodsb
6179 0000AF29 24DF                     <1>         and    al, 0DFh
6180                                    <1>
6181                                    <1> loc_run_check_filename_ext_prg:
6182 0000AF2B 3C50                     <1>         cmp    al, 'P'
6183 0000AF2D 7212                     <1>         jb     short loc_check_prg_fn_retn
6184 0000AF2F 7711                     <1>         ja     short loc_check_prg_fn_stc
6185 0000AF31 AC                       <1>         lodsb
6186 0000AF32 24DF                     <1>         and    al, 0DFh
6187 0000AF34 3C52                     <1>         cmp    al, 'R'
6188 0000AF36 750A                     <1>         jne    short loc_check_prg_fn_stc
6189 0000AF38 AC                       <1>         lodsb
6190 0000AF39 24DF                     <1>         and    al, 0DFh
6191 0000AF3B 3C47                     <1>         cmp    al, 'G'
6192 0000AF3D 7503                     <1>         jne    short loc_check_prg_fn_stc
6193                                    <1>
6194 0000AF3F B050                     <1>         mov    al, 'P'
6195                                    <1> loc_check_prg_fn_retn:
6196 0000AF41 C3                       <1>         retn
6197                                    <1>
6198                                    <1> loc_check_prg_fn_stc:
6199 0000AF42 F9                       <1>         stc
6200 0000AF43 C3                       <1>         retn
6201                                    <1>
6202                                    <1> loc_check_exe_fn_retn:
6203 0000AF44 28C0                     <1>         sub    al, al ; 0
6204 0000AF46 C3                       <1>         retn
6205                                    <1>
6206                                    <1> find_and_list_files:
6207 0000AF47 C3                       <1>         retn
6208                                    <1> set_exec_arguments:
6209 0000AF48 C3                       <1>         retn
6210                                    <1> delete_fs_directory:
6211 0000AF49 31C0                     <1>         xor eax, eax
6212 0000AF4B C3                       <1>         retn
3082                                        %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: 29/12/2017
   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 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
  20                                    <1>         ; 24/01/2016 (TRDOS 386 = TRDOS v2.0)
  21                                    <1>         ; 27/03/2011
  22                                    <1>         ; 09/10/2009
  23                                    <1>         ; INPUT/OUTPUT => none
  24                                    <1>         ; this procedure changes current directory string/text
  25                                    <1>         ; 2005
  26                                    <1>
  27 0000AF4C BE[B3890100]              <1>         mov    esi, PATH_Array
  28                                    <1> change_prompt_dir_str: ; 05/10/2016 (call from 'set_working_path')
  29 0000AF51 BF[5A820100]              <1>         mov    edi, Current_Directory
  30 0000AF56 8A25[54820100]            <1>         mov    ah, [Current_Dir_Level]
  31 0000AF5C E807000000                <1>         call   set_current_directory_string
  32 0000AF61 880D[B5820100]            <1>         mov    [Current_Dir_StrLen], cl
  33                                    <1>
  34 0000AF67 C3                        <1>         retn
  35                                    <1>
  36                                    <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>
0000AF68 57                     <1>       push    edi
0000AF69 80FC00                 <1>       cmp     ah, 0
0000AF6C 7652                   <1>       jna     short pass_write_path
0000AF6E 83C610                 <1>       add     esi, 16
0000AF71 89F3                   <1>       mov     ebx, esi
                                <1> loc_write_path:
0000AF73 B908000000             <1>       mov     ecx, 8
                                <1> path_write_dirname1:
0000AF78 AC                     <1>       lodsb
0000AF79 3C20                   <1>       cmp     al, 20h
0000AF7B 7612                   <1>       jna     short pass_write_dirname1
0000AF7D AA                     <1>       stosb
0000AF7E 81FF[B4820100]         <1>       cmp     edi, End_Of_Current_Dir_Str
0000AF84 733A                   <1>       jnb     short pass_write_path
0000AF86 E2F0                   <1>       loop    path_write_dirname1
0000AF88 803E20                 <1>       cmp     byte [esi], 20h
0000AF8B 7624                   <1>       jna     short pass_write_dirname2
0000AF8D EB0A                   <1>       jmp      short loc_put_dot_cont_ext
                                <1> pass_write_dirname1:
0000AF8F 89DE                   <1>       mov     esi, ebx
0000AF91 83C608                 <1>       add     esi, 8
0000AF94 803E20                 <1>       cmp     byte [esi], 20h
0000AF97 7618                   <1>       jna     short pass_write_dirname2
                                <1> loc_put_dot_cont_ext:
0000AF99 C6072E                 <1>       mov     byte [edi], "."
                                <1>       ;mov    ecx, 3
0000AF9C B103                   <1>       mov     cl, 3
                                <1> loc_check_dir_name_ext:
0000AF9E AC                     <1>       lodsb
0000AF9F 47                     <1>       inc     edi
0000AFA0 3C20                   <1>       cmp     al, 20h
0000AFA2 760D                   <1>       jna     short pass_write_dirname2
0000AFA4 8807                   <1>       mov     [edi], al
0000AFA6 81FF[B4820100]         <1>       cmp     edi, End_Of_Current_Dir_Str
0000AFAC 7312                   <1>       jnb     short pass_write_path
0000AFAE E2EE                   <1>       loop     loc_check_dir_name_ext
0000AFB0 47                     <1>       inc     edi
                                <1> pass_write_dirname2:
0000AFB1 FECC                   <1>       dec     ah
0000AFB3 740B                   <1>       jz       short pass_write_path
0000AFB5 83C310                 <1>       add     ebx, 16
0000AFB8 89DE                   <1>       mov     esi, ebx
0000AFBA C6072F                 <1>       mov     byte [edi],"/"
0000AFBD 47                     <1>       inc     edi
0000AFBE EBB3                   <1>       jmp     short loc_write_path
                                <1> pass_write_path:
0000AFC0 C60700                 <1>       mov     byte [edi], 0
0000AFC3 47                     <1>       inc     edi
0000AFC4 89F9                   <1>       mov     ecx, edi
0000AFC6 5F                     <1>       pop     edi
0000AFC7 29F9                   <1>       sub     ecx, edi
                                <1>       ; ECX = Current Directory String Length
0000AFC9 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:
0000AFCA 80FA00                 <1>       cmp     dl, 0
0000AFCD 7708                   <1>       ja      short loc_get_current_drive_1
0000AFCF 8A15[56820100]         <1>       mov     dl, [Current_Drv]
0000AFD5 EB17                   <1>       jmp     short loc_get_current_drive_2
                                <1> loc_get_current_drive_1:
0000AFD7 FECA                   <1>       dec     dl
0000AFD9 3A15[22380100]         <1>       cmp     dl, [Last_DOS_DiskNo]
0000AFDF 760D                   <1>       jna     short loc_get_current_drive_2
0000AFE1 B80F000000             <1>       mov     eax, 0Fh ; Invalid drive (Drive not ready!)
0000AFE6 F5                     <1>       cmc     ; stc
0000AFE7 C3                     <1>       retn
                                <1>
                                <1> loc_get_current_drive_not_ready_retn:
0000AFE8 5E                     <1>       pop     esi
                                <1>       ;mov    eax, 15
0000AFE9 66B80F00               <1>       mov     ax, 15 ; Drive not ready
0000AFED C3                     <1>       retn
                                <1>
```

```
                                <1> loc_get_current_drive_2:
0000AFEE 31C0                   <1>       xor    eax, eax
0000AFF0 88D4                   <1>       mov    ah, dl
0000AFF2 56                     <1>       push   esi
0000AFF3 BE00010900             <1>       mov    esi, Logical_DOSDisks
0000AFF8 01C6                   <1>       add    esi, eax
0000AFFA 8A06                   <1>       mov    al, [esi+LD_Name]
0000AFFC 3C41                   <1>       cmp    al, 'A'
0000AFFE 72E8                   <1>       jb     short loc_get_current_drive_not_ready_retn
                                <1>
0000B000 8A667F                 <1>       mov    ah, [esi+LD_CDirLevel]
0000B003 08E4                   <1>       or     ah, ah
0000B005 7506                   <1>       jnz    short loc_get_current_drive_3
                                <1>
                                <1>       ;xor   ah, ah ; mov ah, 0
0000B007 8826                   <1>       mov    [esi], ah
0000B009 31C9                   <1>       xor    ecx, ecx
0000B00B EB1C                   <1>       jmp    short loc_get_current_drive_4
                                <1>
                                <1> loc_get_current_drive_3:
0000B00D BF[B3890100]           <1>         mov     edi, PATH_Array
0000B012 57                     <1>       push   edi
0000B013 81C680000000           <1>       add    esi, LD_CurrentDirectory
0000B019 B920000000             <1>       mov    ecx, 32
0000B01E F3A5                   <1>       rep    movsd
0000B020 5E                     <1>       pop    esi ; Path Array Address
0000B021 5F                     <1>       pop    edi ; pushed esi (current dir buffer offset)
                                <1>       ;
0000B022 E841FFFFFF             <1>       call   set_current_directory_string
0000B027 89FE                   <1>       mov    esi, edi
                                <1>
                                <1> loc_get_current_drive_4:
0000B029 30C0                   <1>       xor    al, al
0000B02B C3                     <1>       retn
                                <1>
                                <1> change_current_directory:
                                <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>
0000B02C 8825[418A0100]         <1>       mov    [CD_COMMAND], ah
0000B032 803E2F                 <1>       cmp    byte [esi], '/'
0000B035 7505                   <1>       jne    short loc_ccd_cdir_level
0000B037 46                     <1>       inc    esi
0000B038 30C0                   <1>       xor    al, al
0000B03A EB05                   <1>       jmp    short loc_ccd_parse_path_name
                                <1> loc_ccd_cdir_level:
0000B03C A0[54820100]           <1>       mov    al, [Current_Dir_Level]
                                <1> loc_ccd_parse_path_name:
0000B041 88C4                   <1>       mov    ah, al
0000B043 BF[B3890100]           <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>
0000B048 0FB6C8                 <1>       movzx  ecx, al
0000B04B FEC1                   <1>       inc    cl
0000B04D C0E104                 <1>       shl    cl, 4
0000B050 01CF                   <1>       add    edi, ecx
0000B052 B107                   <1>       mov    cl, 7
0000B054 28C1                   <1>       sub    cl, al
0000B056 C0E102                 <1>       shl    cl, 2
0000B059 89C3                   <1>       mov    ebx, eax
0000B05B 31C0                   <1>       xor    eax, eax ; 0
0000B05D F3AB                   <1>       rep    stosd
0000B05F 89D8                   <1>       mov    eax, ebx
                                <1>
0000B061 BF[B3890100]           <1>       mov    edi, PATH_Array
                                <1>
0000B066 803E20                 <1>       cmp    byte [esi], 20h
0000B069 F5                     <1>       cmc
0000B06A 7305                   <1>       jnc    short pass_ccd_parse_dir_name
                                <1>
                                <1>              ; ESI = Path name
                                <1>              ; AL = CCD_Level
0000B06C E872010000             <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:
0000B071 9C                     <1>       pushf
```

```
                        <1>
                        <1>       ;mov   [CCD_Level], al
                        <1>         ;mov [Last_Dir_Level], ah
0000B072 66A3[378A0100] <1>       mov    [CCD_Level], ax
                        <1>
0000B078 31DB           <1>       xor    ebx, ebx
0000B07A 8A3D[56820100] <1>       mov    bh, [Current_Drv]
0000B080 BE00010900     <1>       mov    esi, Logical_DOSDisks
0000B085 01DE           <1>       add    esi, ebx
                        <1>
0000B087 9D             <1>       popf
0000B088 720A           <1>       jc     short loc_ccd_bad_path_name_retn
                        <1>
0000B08A 8935[338A0100] <1>       mov    [CCD_DriveDT], esi
                        <1>
0000B090 3C07           <1>       cmp    al, 7
0000B092 7209           <1>       jb     short loc_ccd_load_child_dir
                        <1>
                        <1> loc_ccd_bad_path_name_retn:
0000B094 87F7           <1>       xchg   esi, edi
0000B096 B813000000     <1>       mov    eax, 19 ; Bad directory/path name
0000B09B F9             <1>       stc
                        <1> loc_ccd_retn_p:
0000B09C C3             <1>       retn
                        <1>
                        <1> loc_ccd_load_child_dir:
                        <1>       ; AL = CCD_Level
0000B09D 08C0           <1>       or     al, al
0000B09F 7468           <1>       jz     short loc_ccd_load_root_dir
                        <1>
0000B0A1 6689C1         <1>       mov    cx, ax
0000B0A4 C0E004         <1>       shl    al, 4
0000B0A7 0FB6F0         <1>       movzx  esi, al
0000B0AA 01FE           <1>       add    esi, edi  ; offset PATH_Array
                        <1>
0000B0AC 8B460C         <1>       mov    eax, [esi+12]
0000B0AF 38E9           <1>       cmp    cl, ch
0000B0B1 0F84FA000000   <1>         je      loc_ccd_load_sub_directory
0000B0B7 A3[50820100]   <1>       mov    [Current_Dir_FCluster], eax
                        <1>
                        <1> loc_ccd_load_child_dir_next:
0000B0BC 83C610         <1>       add    esi, 16 ; DOS DirEntry Format FileName Address
                        <1>
                        <1>       ; Directory attribute : 10h
0000B0BF B010           <1>       mov    al, 00010000b ; 10h (Attrib AND mask)
                        <1>       ;mov   ah, 11001000b ; C8h
                        <1>       ; Volume name attribute: 8h
0000B0C1 B408           <1>       mov    ah, 00001000b ; 08h (Attrib NAND, AND --> zero mask)
                        <1>
0000B0C3 6631C9         <1>       xor    cx, cx
0000B0C6 E8B5010000     <1>       call   locate_current_dir_file
0000B0CB 7353           <1>       jnc    short loc_ccd_set_dir_cluster_ptr
                        <1>
                        <1>        ; 19/02/2016
                        <1>       ;mov   edi, [CCD_DriveDT]
0000B0CD 8A25[378A0100] <1>       mov    ah, [CCD_Level]
0000B0D3 803D[418A0100]CD <1>     cmp    byte [CD_COMMAND], 0CDh ;'CD' command or another
0000B0DA 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.)
0000B0DC 88E1           <1>       mov    cl, ah
0000B0DE 50             <1>       push   eax
0000B0DF E8E3000000     <1>       call   loc_ccd_save_current_dir
0000B0E4 58             <1>       pop    eax
                        <1> loc_ccd_load_child_dir_err:
0000B0E5 3C03           <1>       cmp    al, 3  ; AL = 2 => File not found error
0000B0E7 7202           <1>       jb     short loc_ccd_path_not_found_retn
0000B0E9 F9             <1>       stc
0000B0EA C3             <1>       retn
                        <1>
                        <1> loc_ccd_path_not_found_retn:
0000B0EB B003           <1>       mov    al, 3  ; Path not found
0000B0ED C3             <1>       retn
                        <1>
                        <1> loc_ccd_load_FAT_root_dir:
0000B0EE 803D[55820100]02 <1>     cmp    byte [Current_FATType], 2
0000B0F5 776B           <1>       ja     short loc_ccd_load_FAT32_root_dir
                        <1>
                        <1>       ;mov   esi, [CCD_DriveDT]
                        <1>       ;push  esi
0000B0F7 E8B51D0000     <1>       call   load_FAT_root_directory
                        <1>       ;pop   edi ; Dos Drv Description Table
                        <1>
0000B0FC 89F7           <1>       mov    edi, esi
0000B0FE BE[B3890100]   <1>       mov    esi, PATH_Array
0000B103 7297           <1>       jc     short loc_ccd_retn_p
                        <1>
0000B105 31C0           <1>       xor    eax, eax
0000B107 EB78           <1>         jmp  short loc_ccd_set_cdfc
                        <1>
                        <1> loc_ccd_load_root_dir:
0000B109 803D[55820100]01 <1>     cmp    byte [Current_FATType], 1
0000B110 73DC           <1>       jnb    short loc_ccd_load_FAT_root_dir
                        <1>
                        <1> loc_ccd_load_FS_root_dir:
0000B112 E8611E0000     <1>       call   load_FS_root_directory
0000B117 EB5C           <1>       jmp    short pass_ccd_load_FAT_sub_directory
                        <1>
                        <1> loc_ccd_load_FS_sub_directory_next:
0000B119 E85B1E0000     <1>       call   load_FS_sub_directory
0000B11E EB1F           <1>       jmp    short pass_ccd_set_dir_cluster_ptr
                        <1>
                        <1> loc_ccd_set_dir_cluster_ptr:
```

```
                                <1>       ; EDI = Directory Entry
0000B120 668B4714               <1>       mov   ax, [edi+20] ; First Cluster High Word
0000B124 C1E010                 <1>       shl   eax, 16
0000B127 668B471A               <1>       mov   ax, [edi+26] ; First Cluster Low Word
                                <1>
0000B12B 8B35[338A0100]         <1>       mov   esi, [CCD_DriveDT]
0000B131 803D[55820100]01       <1>       cmp   byte [Current_FATType], 1
0000B138 72DF                   <1>       jb    short loc_ccd_load_FS_sub_directory_next
                                <1>       ;push  esi
0000B13A E8FD1D0000             <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
0000B13F BE[B3890100]           <1>       mov   esi, PATH_Array
0000B144 7264                   <1>       jc    short loc_ccd_retn_c
                                <1>
0000B146 A1[81890100]           <1>       mov   eax, [DirBuff_Cluster]
                                <1>
0000B14B FE05[378A0100]         <1>       inc   byte [CCD_Level]
0000B151 0FB61D[378A0100]       <1>       movzx ebx, byte [CCD_Level]
0000B158 C0E304                 <1>       shl   bl, 4 ; * 16 (<= 128)
0000B15B 01DE                   <1>       add   esi, ebx ; 19/02/2016
0000B15D 89460C                 <1>       mov   [esi+12], eax
0000B160 EB1F                   <1>       jmp   short loc_ccd_set_cdfc
                                <1>
                                <1> loc_ccd_load_FAT32_root_dir:
0000B162 BE[B3890100]           <1>       mov   esi, PATH_Array
0000B167 8B460C                 <1>       mov   eax, [esi+12]
0000B16A 8B35[338A0100]         <1>       mov   esi, [CCD_DriveDT]
                                <1>
                                <1> loc_ccd_load_FAT_sub_directory:
                                <1>       ;push  esi
0000B170 E8C71D0000             <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
0000B175 BE[B3890100]           <1>       mov   esi, PATH_Array
0000B17A 722E                   <1>       jc    short loc_ccd_retn_c
                                <1>
0000B17C A1[81890100]           <1>       mov   eax, [DirBuff_Cluster]
                                <1>
                                <1> loc_ccd_set_cdfc:
0000B181 8A0D[378A0100]         <1>       mov   cl, [CCD_Level]
0000B187 880D[54820100]         <1>       mov   [Current_Dir_Level], cl
0000B18D A3[50820100]           <1>       mov   [Current_Dir_FCluster], eax
                                <1>
0000B192 8A2D[388A0100]         <1>       mov   ch, [Last_Dir_Level]
0000B198 38E9                   <1>       cmp   cl, ch
0000B19A 0F821CFFFFFF           <1>       jb    loc_ccd_load_child_dir_next
                                <1>
0000B1A0 803D[418A0100]CD       <1>       cmp   byte [CD_COMMAND], 0CDh ;'CD' command or another
0000B1A7 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>
0000B1A9 F8                     <1>       clc
                                <1>
                                <1> loc_ccd_retn_c:
0000B1AA 8B3D[338A0100]         <1>       mov   edi, [CCD_DriveDT]
0000B1B0 C3                     <1>       retn
                                <1>
                                <1> loc_ccd_load_sub_directory:
0000B1B1 8B35[338A0100]         <1>       mov   esi, [CCD_DriveDT]
0000B1B7 803D[55820100]01       <1>       cmp   byte [Current_FATType], 1
0000B1BE 73B0                   <1>       jnb   short loc_ccd_load_FAT_sub_directory
0000B1C0 E8B41D0000             <1>       call  load_FS_sub_directory
0000B1C5 EBAE                   <1>       jmp   short pass_ccd_load_FAT_sub_directory
                                <1>
                                <1> loc_ccd_save_current_dir:
0000B1C7 BE[B3890100]           <1>       mov   esi, PATH_Array ; 19/02/2016
0000B1CC 8B3D[338A0100]         <1>       mov   edi, [CCD_DriveDT]
0000B1D2 57                     <1>       push  edi
0000B1D3 83C77F                 <1>         add     edi, LD_CDirLevel
0000B1D6 880F                   <1>       mov   [edi], cl
0000B1D8 47                     <1>       inc   edi ; LD_CurrentDirectory
0000B1D9 56                     <1>       push  esi
                                <1>       ;mov  ecx, 32  ; always < 65536 (in this procedure)
0000B1DA 66B92000               <1>       mov   cx, 32
0000B1DE F3A5                   <1>       rep   movsd
                                <1>       ; Current directory has been saved to
                                <1>       ; the DOS drive description table, cdir area !
0000B1E0 5E                     <1>       pop   esi  ; PATH_Array
0000B1E1 5F                     <1>       pop   edi  ; Dos Drv Description Table
                                <1>
0000B1E2 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
0000B1E3 88C4           <1>       mov    ah, al
0000B1E5 66A3[D88A0100] <1>       mov    [PATH_CDLevel], ax
                        <1> repeat_ppdn_check_slash:
0000B1EB AC             <1>       lodsb
0000B1EC 3C2F           <1>       cmp    al, '/'
0000B1EE 74FB           <1>       je     short repeat_ppdn_check_slash
0000B1F0 3C21           <1>       cmp    al, 21h
0000B1F2 7219           <1>       jb     short loc_ppdn_retn
0000B1F4 57             <1>       push   edi
                        <1> loc_ppdn_get_dir_name:
0000B1F5 B90C000000     <1>       mov    ecx, 12
0000B1FA BF[DA8A0100]   <1>       mov    edi, Dir_File_Name
                        <1> repeat_ppdn_get_dir_name:
0000B1FF AA             <1>       stosb
0000B200 AC             <1>       lodsb
0000B201 3C2F           <1>       cmp    al, '/'
0000B203 740A           <1>       je     short loc_check_level_dot_conv_dir_name
0000B205 3C20           <1>       cmp    al, 20h
0000B207 7605           <1>       jna    short loc_ppdn_end_of_path_scan
0000B209 E2F4           <1>       loop   repeat_ppdn_get_dir_name
0000B20B 5F             <1>       pop    edi
0000B20C F9             <1>       stc
                        <1> loc_ppdn_retn:
0000B20D C3             <1>       retn
                        <1>
                        <1> loc_ppdn_end_of_path_scan:
0000B20E 4E             <1>       dec    esi
                        <1> loc_check_level_dot_conv_dir_name:
0000B20F 31C0           <1>       xor    eax, eax
0000B211 AA             <1>       stosb
0000B212 89F3           <1>       mov    ebx, esi
0000B214 BE[DA8A0100]   <1>       mov    esi, Dir_File_Name
0000B219 AC             <1>       lodsb
                        <1> repeat_ppdn_name_check_dot:
0000B21A 3C2E           <1>       cmp    al, '.'
0000B21C 7509           <1>       jne    short loc_ppdn_convert_sub_dir_name
                        <1> repeat_ppdn_name_dot_dot:
0000B21E AC             <1>       lodsb
0000B21F 3C2E           <1>       cmp    al, '.'
0000B221 743E           <1>       je     short loc_ppdn_dot_dot
0000B223 3C21           <1>       cmp    al, 21h
0000B225 7226           <1>       jb     short pass_ppdn_convert_sub_dir_name
                        <1> loc_ppdn_convert_sub_dir_name:
0000B227 8A25[D98A0100] <1>       mov    ah, [PATH_Level]
0000B22D 80FC07         <1>       cmp    ah, 7
0000B230 731B           <1>       jnb    short pass_ppdn_convert_sub_dir_name
0000B232 FEC4           <1>       inc    ah
0000B234 8825[D98A0100] <1>       mov    [PATH_Level], ah
0000B23A BE[DA8A0100]   <1>       mov    esi, Dir_File_Name
                        <1>       ;mov   edi, [PATH_Array_Ptr]
0000B23F B010           <1>       mov    al, 16
0000B241 F6E4           <1>       mul    ah
0000B243 8B3C24         <1>       mov    edi, [esp]
                        <1>       ;push  edi
0000B246 01C7           <1>       add    edi, eax
0000B248 E82A030000     <1>       call   convert_file_name
                        <1>       ;pop   edi
                        <1> pass_ppdn_convert_sub_dir_name:
0000B24D 89DE           <1>       mov    esi, ebx
                        <1> repeat_ppdn_check_last_slash:
0000B24F AC             <1>       lodsb
0000B250 3C2F           <1>       cmp    al, '/'
0000B252 74FB           <1>       je     short repeat_ppdn_check_last_slash
0000B254 3C21           <1>       cmp    al, 21h
0000B256 739D           <1>       jnb    short loc_ppdn_get_dir_name
                        <1> end_of_parse_dir_name:
0000B258 5F             <1>       pop    edi
0000B259 F5             <1>       cmc
                        <1>       ;mov   al, [PATH_CDLevel]
                        <1>       ;mov   ah, [PATH_Level]
0000B25A 66A1[D88A0100] <1>       mov    ax, [PATH_CDLevel]
0000B260 C3             <1>       retn
                        <1>
                        <1> loc_ppdn_dot_dot:
0000B261 AC             <1>       lodsb
0000B262 3C21           <1>       cmp    al, 21h
0000B264 73F2           <1>       jnb    short end_of_parse_dir_name
                        <1> loc_ppdn_dot_dot_prev_level:
0000B266 66A1[D88A0100] <1>       mov    ax, [PATH_CDLevel]
0000B26C 80EC01         <1>       sub    ah, 1
0000B26F 80D400         <1>       adc    ah, 0
0000B272 38E0           <1>       cmp    al, ah
0000B274 7602           <1>       jna    short pass_ppdn_set_al_to_ah
0000B276 88E0           <1>       mov    al, ah
                        <1> pass_ppdn_set_al_to_ah:
0000B278 66A3[D88A0100] <1>       mov    [PATH_CDLevel], ax
0000B27E 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>
0000B280 8935[3B8A0100]         <1>       mov    [CDLF_FNAddress], esi
0000B286 66A3[398A0100]         <1>       mov    [CDLF_AttributesMask], ax
0000B28C 66890D[3F8A0100]       <1>       mov    [CDLF_DEType], cx
                                <1>
0000B293 31DB                   <1>       xor    ebx, ebx
0000B295 881D[508A0100]         <1>       mov    [PreviousAttr], bl ; 0  ; 13/02/2016
                                <1>
0000B29B 8A3D[56820100]         <1>       mov    bh, [Current_Drv]
0000B2A1 381D[7C890100]         <1>       cmp    byte [DirBuff_ValidData], bl ; 0
0000B2A7 761D                   <1>       jna    short loc_lcdf_reload_current_dir2
0000B2A9 8A1D[7A890100]         <1>         mov     bl, [DirBuff_DRV]
0000B2AF 80EB41                 <1>       sub    bl, 'A'
0000B2B2 38DF                   <1>       cmp    bh, bl
0000B2B4 750E                   <1>       jne    short loc_lcdf_reload_current_dir1
0000B2B6 8B15[81890100]         <1>       mov    edx, [DirBuff_Cluster]
0000B2BC 3B15[50820100]         <1>       cmp    edx, [Current_Dir_FCluster]
0000B2C2 7412                   <1>       je     short loc_cdir_locatefile_search
                                <1>
                                <1> loc_lcdf_reload_current_dir1:
0000B2C4 30DB                   <1>       xor    bl, bl
                                <1> loc_lcdf_reload_current_dir2:
0000B2C6 89DE                   <1>       mov    esi, ebx
0000B2C8 81C600010900           <1>         add     esi, Logical_DOSDisks
0000B2CE E874000000             <1>       call   reload_current_directory
0000B2D3 735D                   <1>       jnc    short loc_locatefile_search_again
0000B2D5 C3                     <1>       retn
                                <1>
                                <1> loc_cdir_locatefile_search:
0000B2D6 31DB                   <1>       xor    ebx, ebx
0000B2D8 55                     <1>       push   ebp ; 20/11/2017
0000B2D9 E8A6000000             <1>       call   find_directory_entry
0000B2DE 5D                     <1>       pop    ebp ; 20/11/2017
0000B2DF 7349                   <1>       jnc    short loc_cdir_locate_file_retn
                                <1>
                                <1> loc_locatefile_check_stc_reason:
0000B2E1 08ED                   <1>       or     ch, ch
0000B2E3 7444                   <1>       jz     short loc_cdir_locate_file_stc_retn
                                <1>
                                <1> loc_locatefile_check_next_entryblock:
0000B2E5 8A3D[56820100]         <1>       mov    bh, [Current_Drv]
0000B2EB 28DB                   <1>       sub    bl, bl
0000B2ED 0FB7F3                 <1>       movzx  esi, bx
0000B2F0 81C600010900           <1>         add     esi, Logical_DOSDisks
                                <1>
0000B2F6 803D[54820100]00       <1>       cmp    byte [Current_Dir_Level], 0
0000B2FD 760A                   <1>       jna    short loc_locatefile_check_FAT_type
                                <1>
0000B2FF 803D[55820100]01       <1>       cmp    byte [Current_FATType], 1
0000B306 730A                   <1>       jnb    short loc_locatefile_load_subdir_cluster
0000B308 C3                     <1>       retn
                                <1>
                                <1> loc_locatefile_check_FAT_type:
0000B309 803D[55820100]03       <1>       cmp    byte [Current_FATType], 3
0000B310 7218                   <1>       jb     short loc_cdir_locate_file_retn
                                <1>
                                <1> loc_locatefile_load_subdir_cluster:
0000B312 A1[81890100]           <1>       mov    eax, [DirBuff_Cluster]
0000B317 E83A1A0000             <1>       call   get_next_cluster
0000B31C 730D                   <1>       jnc    short loc_locatefile_next_cluster
0000B31E 09C0                   <1>       or     eax, eax
0000B320 7507                   <1>       jnz    short loc_locatefile_drive_not_ready_read_err
0000B322 F9                     <1>       stc
                                <1> loc_locatefile_file_notfound:
0000B323 B802000000             <1>       mov    eax, 2 ; File/Directory/VolName not found
0000B328 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:
0000B329 F5                     <1>       cmc ;stc
                                <1> loc_cdir_locate_file_retn:
0000B32A C3                     <1>       retn
                                <1>
                                <1> loc_locatefile_next_cluster:
0000B32B E80C1C0000             <1>       call   load_FAT_sub_directory
```

```
                            <1>       ;jc    short loc_locatefile_drive_not_ready_read_err
0000B330 72F8               <1>       jc     short loc_cdir_locate_file_retn
                            <1>
                            <1> loc_locatefile_search_again:
0000B332 8B35[3B8A0100]     <1>       mov    esi, [CDLF_FNAddress]
0000B338 66A1[398A0100]     <1>       mov    ax, [CDLF_AttributesMask]
0000B33E 668B0D[3F8A0100]   <1>       mov    cx, [CDLF_DEType]
0000B345 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]
0000B347 A0[55820100]       <1>       mov    al, [Current_FATType]
0000B34C 3C02               <1>       cmp    al, 2
0000B34E 7729               <1>       ja     short loc_reload_FAT_sub_directory
0000B350 8A25[54820100]     <1>       mov    ah, [Current_Dir_Level]
0000B356 08C0               <1>       or     al, al
0000B358 740A               <1>       jz     short loc_reload_FS_directory
0000B35A 08E4               <1>       or     ah, ah
0000B35C 751B               <1>       jnz    short loc_reload_FAT_sub_directory
                            <1> loc_reload_FAT_12_16_root_directory:
0000B35E E84E1B0000         <1>       call   load_FAT_root_directory
0000B363 C3                 <1>       retn
                            <1> loc_reload_FS_directory:
0000B364 20E4               <1>       and    ah, ah
0000B366 7506               <1>       jnz    short loc_reload_FS_sub_directory
                            <1> loc_reload_FS_root_directory:
0000B368 E80B1C0000         <1>       call   load_FS_root_directory
0000B36D C3                 <1>       retn
                            <1> loc_reload_FS_sub_directory:
0000B36E A1[50820100]       <1>       mov    eax, [Current_Dir_FCluster]
0000B373 E8011C0000         <1>       call   load_FS_sub_directory
0000B378 C3                 <1>       retn
                            <1> loc_reload_FAT_sub_directory:
0000B379 A1[50820100]       <1>       mov    eax, [Current_Dir_FCluster]
0000B37E E8B91B0000         <1>       call   load_FAT_sub_directory
0000B383 C3                 <1>       retn
                            <1>
                            <1> find_directory_entry:
                            <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>
0000B384 663B1D[7F890100]   <1>       cmp    bx, [DirBuff_LastEntry]
0000B38B 0F8739010000       <1>         ja      loc_ffde_stc_retn_255
                            <1>
                            <1>       ;mov    [DirBuff_CurrentEntry], bx
                            <1>
0000B391 BF00000800         <1>       mov    edi, Directory_Buffer
0000B396 66A3[4C8A0100]     <1>       mov    [FDE_AttrMask], ax
                            <1>
0000B39C 29C0               <1>       sub    eax, eax
                            <1>
                            <1>       ;;mov  [PreviousAttr], al ; 0 ;; 13/02/2016
0000B39E 66A3[4E8A0100]     <1>       mov    [AmbiguousFileName], ax ; 0
                            <1>
0000B3A4 6689D8             <1>       mov    ax, bx
0000B3A7 66C1E005           <1>       shl    ax, 5 ; ; * 32 ; Directory entry size
0000B3AB 01C7               <1>       add     edi, eax
                            <1>
0000B3AD 08ED               <1>       or     ch, ch
```

```
0000B3AF 0F852C010000          <1>         jnz     loc_find_free_deleted_entry_0
                               <1>
0000B3B5 08C9                  <1>       or      cl, cl
0000B3B7 0F850D010000          <1>         jnz     loc_ffde_stc_retn_255
                               <1>
                               <1> check_find_dir_entry:
0000B3BD 66A1[4C8A0100]        <1>       mov     ax, [FDE_AttrMask]
0000B3C3 8A2F                  <1>       mov     ch, [edi]
0000B3C5 80FD00                <1>       cmp     ch, 0 ; Is it never used entry?
0000B3C8 0F86FF000000          <1>       jna     loc_find_direntry_stc_retn
0000B3CE 56                    <1>       push    esi
0000B3CF 8A570B                <1>       mov     dl, [edi+0Bh] ; File attributes
0000B3D2 80FDE5                <1>       cmp     ch, 0E5h ; Is it a deleted file?
0000B3D5 746D                  <1>       je      short loc_find_dir_next_entry_prevdeleted
                               <1>
0000B3D7 80FA0F                <1>       cmp     dl, 0Fh ; longname sub component check
0000B3DA 7505                  <1>       jne     short loc_check_attributes_mask
0000B3DC E8ED010000            <1>       call    save_longname_sub_component
                               <1>
                               <1> loc_check_attributes_mask:
0000B3E1 88C6                  <1>       mov     dh, al
0000B3E3 20D6                  <1>       and     dh, dl
0000B3E5 38F0                  <1>       cmp     al, dh
0000B3E7 0F85BA000000          <1>         jne     loc_find_dir_next_entry
0000B3ED 20D4                  <1>       and     ah, dl
0000B3EF 0F85B2000000          <1>         jnz     loc_find_dir_next_entry
0000B3F5 80FA0F                <1>       cmp     dl, 0Fh
0000B3F8 751A                  <1>       jne     short pass_direntry_attr_check
                               <1>
0000B3FA 3C0F                  <1>       cmp     al, 0Fh ; AL = 0Fh -> find long name
0000B3FC 0F85A5000000          <1>         jne     loc_find_dir_next_entry
                               <1>
0000B402 5E                    <1>       pop     esi
0000B403 6631C0                <1>       xor     ax, ax
0000B406 8A35[508A0100]        <1>       mov     dh, [PreviousAttr]
0000B40C 66891D[7D890100]      <1>       mov     [DirBuff_CurrentEntry], bx
0000B413 C3                    <1>       retn
                               <1>
                               <1> pass_direntry_attr_check:
0000B414 89FD                  <1>       mov     ebp, edi ; 14/02/2016
0000B416 B908000000            <1>       mov     ecx, 8
                               <1> loc_lodsb_find_dir:
0000B41B AC                    <1>       lodsb
0000B41C 3C2A                  <1>       cmp     al, '*'
0000B41E 7508                  <1>       jne     short pass_fde_ambiguous1_check
0000B420 FE05[4F8A0100]        <1>         inc     byte [AmbiguousFileName+1]
0000B426 EB28                  <1>       jmp     short loc_check_direntry_extension
                               <1>
                               <1> pass_fde_ambiguous1_check:
0000B428 3C3F                  <1>       cmp     al, '?'
0000B42A 750D                  <1>       jne     short pass_fde_ambiguous2_check
0000B42C FE05[4E8A0100]        <1>       inc     byte [AmbiguousFileName]
0000B432 803F20                <1>       cmp     byte [edi], 20h
0000B435 764E                  <1>       jna     short loc_find_dir_next_entry_ebp
0000B437 EB14                  <1>       jmp     short loc_scasb_find_dir_inc_di
                               <1>
                               <1> pass_fde_ambiguous2_check:
0000B439 3C20                  <1>       cmp     al, 20h
0000B43B 750C                  <1>       jne     short loc_scasb_find_dir
0000B43D 803F20                <1>       cmp     byte [edi], 20h
0000B440 7543                  <1>       jne     short loc_find_dir_next_entry_ebp
0000B442 EB0C                  <1>       jmp     short loc_check_direntry_extension
                               <1>
                               <1> loc_find_dir_next_entry_prevdeleted:
0000B444 80CA80                <1>       or      dl, 80h  ; Bit 7 -> deleted entry sign
0000B447 EB5E                  <1>       jmp     short loc_find_dir_next_entry
                               <1>
                               <1> loc_scasb_find_dir:
0000B449 3A07                  <1>       cmp     al, [edi]
0000B44B 7538                  <1>       jne     short loc_find_dir_next_entry_ebp
                               <1> loc_scasb_find_dir_inc_di:
0000B44D 47                    <1>       inc     edi
0000B44E E2CB                  <1>       loop    loc_lodsb_find_dir
                               <1>
                               <1> loc_check_direntry_extension:
0000B450 BE08000000            <1>       mov     esi, 8
0000B455 89F7                  <1>       mov     edi, esi ; 8
0000B457 033424                <1>       add     esi, [esp] ; Sub Dir or File Name Address
0000B45A 01EF                  <1>       add     edi, ebp
0000B45C B103                  <1>       mov     cl, 3
                               <1> loc_lodsb_find_dir_ext:
0000B45E AC                    <1>       lodsb
0000B45F 3C2A                  <1>       cmp     al, '*'
0000B461 7508                  <1>       jne     short pass_fde_ambiguous3_check
0000B463 FE05[4F8A0100]        <1>       inc     byte [AmbiguousFileName+1]
0000B469 EB1E                  <1>       jmp     short loc_find_dir_proper_direntry
                               <1>
                               <1> pass_fde_ambiguous3_check:
0000B46B 3C3F                  <1>       cmp     al, '?'
0000B46D 750D                  <1>       jne     short pass_fde_ambiguous4_check
0000B46F FE05[4E8A0100]        <1>       inc     byte [AmbiguousFileName]
0000B475 803F20                <1>       cmp     byte [edi], 20h
0000B478 760B                  <1>       jna     short loc_find_dir_next_entry_ebp
0000B47A EB49                  <1>       jmp     short loc_scasb_find_dir_ext_inc_di
                               <1>
                               <1> pass_fde_ambiguous4_check:
0000B47C 3C20                  <1>       cmp     al, 20h
0000B47E 7541                  <1>       jne     short loc_scasb_find_dir_ext
0000B480 803F20                <1>       cmp     byte [edi], 20h
0000B483 7404                  <1>       je      short loc_find_dir_proper_direntry
                               <1>
                               <1> loc_find_dir_next_entry_ebp:
0000B485 89EF                  <1>       mov     edi, ebp ; 14/02/2016
0000B487 EB1E                  <1>       jmp     short loc_find_dir_next_entry
                               <1>
```

```
                                <1> loc_find_dir_proper_direntry:
0000B489 30C9                   <1>       xor    cl, cl
                                <1> loc_find_dir_proper_direntry_1:
0000B48B 5E                     <1>       pop    esi
0000B48C 89EF                   <1>         mov     edi, ebp
0000B48E 8A2F                   <1>       mov    ch, [edi]
0000B490 8A570B                 <1>       mov     dl, [edi+0Bh] ; Dir entry attributes
0000B493 66A1[4E8A0100]         <1>       mov    ax, [AmbiguousFileName]
                                <1> loc_find_dir_proper_direntry_2:
0000B499 8A35[508A0100]         <1>       mov     dh, [PreviousAttr]
0000B49F 66891D[7D890100]       <1>       mov    [DirBuff_CurrentEntry], bx
0000B4A6 C3                     <1>       retn
                                <1>
                                <1> loc_find_dir_next_entry:
0000B4A7 8815[508A0100]         <1>       mov    byte [PreviousAttr], dl ; LongName check
                                <1> loc_find_dir_next_entry_1:
0000B4AD 5E                     <1>       pop    esi
0000B4AE 83C720                 <1>       add    edi, 32
                                <1>       ;inc   word [DirBuff_EntryCounter]
0000B4B1 6643                   <1>       inc    bx
0000B4B3 663B1D[7F890100]       <1>       cmp    bx, [DirBuff_LastEntry]
0000B4BA 770E                   <1>       ja     short loc_ffde_stc_retn_255
0000B4BC E9FCFEFFFF             <1>         jmp     check_find_dir_entry
                                <1>
                                <1> loc_scasb_find_dir_ext:
0000B4C1 3A07                   <1>       cmp    al, [edi]
0000B4C3 75C0                   <1>       jne    short loc_find_dir_next_entry_ebp
                                <1> loc_scasb_find_dir_ext_inc_di:
0000B4C5 47                     <1>       inc    edi
0000B4C6 E296                   <1>       loop    loc_lodsb_find_dir_ext
0000B4C8 EBC1                   <1>       jmp    short loc_find_dir_proper_direntry_1
                                <1>
                                <1> loc_ffde_stc_retn_255:
                                <1>       ;mov   cx, 0FFFFh
0000B4CA 31C9                   <1>       xor    ecx, ecx
0000B4CC 49                     <1>       dec    ecx ; 0FFFFFFFFh
                                <1>       ;xor   eax, eax
                                <1> loc_find_direntry_stc_retn:
                                <1> loc_check_ffde_retn_1:
                                <1>       ;mov   ax, 2
0000B4CD B802000000             <1>       mov    eax, 2 ; File Not Found
0000B4D2 8A35[508A0100]         <1>       mov    dh, [PreviousAttr]
0000B4D8 66891D[7D890100]       <1>       mov    [DirBuff_CurrentEntry], bx
0000B4DF F9                     <1>       stc
0000B4E0 C3                     <1>       retn
                                <1>
                                <1> loc_find_free_deleted_entry_0:
0000B4E1 66A1[4C8A0100]         <1>       mov    ax, [FDE_AttrMask]
0000B4E7 8A2F                   <1>       mov    ch, [edi]
0000B4E9 8A570B                 <1>       mov    dl, [edi+0Bh] ; File attributes
0000B4EC 08C9                   <1>       or     cl, cl
0000B4EE 7407                   <1>       jz     short loc_check_ffde_0_repeat
                                <1>       ;cmp   cl, 0E5h
                                <1>       ;je    short pass_loc_check_ffde_0_err
0000B4F0 80F9FF                 <1>       cmp    cl, 0FFh
0000B4F3 7432                   <1>       je     short loc_find_free_deleted_entry_1
0000B4F5 EB4D                   <1>       jmp    short pass_loc_check_ffde_0_err
                                <1>
                                <1> loc_check_ffde_0_repeat:
0000B4F7 08ED                   <1>       or     ch, ch
0000B4F9 7511                   <1>       jnz    short loc_check_ffde_0_next
                                <1>
                                <1> loc_check_ffde_retn_2:
0000B4FB 6629C0                 <1>       sub    ax, ax
0000B4FE 8A35[508A0100]         <1>       mov    dh, [PreviousAttr]
0000B504 66891D[7D890100]       <1>       mov    [DirBuff_CurrentEntry], bx
0000B50B C3                     <1>       retn
                                <1>
                                <1> loc_check_ffde_0_next:
0000B50C 6643                   <1>       inc    bx
0000B50E 83C720                 <1>       add    edi, 32
                                <1>       ;inc   word [DirBuff_EntryCounter]
                                <1>
0000B511 663B1D[7F890100]       <1>         cmp  bx, [DirBuff_LastEntry]
0000B518 77B0                   <1>       ja     short loc_ffde_stc_retn_255
0000B51A 8815[508A0100]         <1>       mov    [PreviousAttr], dl
0000B520 8A2F                   <1>       mov    ch, [edi]
0000B522 8A570B                 <1>       mov    dl, [edi+0Bh] ; file attributes
0000B525 EBD0                   <1>       jmp    short loc_check_ffde_0_repeat
                                <1>
                                <1> loc_find_free_deleted_entry_1:
0000B527 28D2                   <1>       sub    dl, dl
                                <1> loc_find_free_deleted_entry_2:
0000B529 20ED                   <1>       and    ch, ch
0000B52B 74CE                   <1>       jz     short loc_check_ffde_retn_2
0000B52D 80FDE5                 <1>       cmp    ch, 0E5h
0000B530 74C9                   <1>       je     short loc_check_ffde_retn_2
0000B532 6643                   <1>       inc    bx
0000B534 83C720                 <1>       add    edi, 32
0000B537 663B1D[7F890100]       <1>       cmp    bx, [DirBuff_LastEntry]
0000B53E 778A                   <1>       ja     short loc_ffde_stc_retn_255
0000B540 8A2F                   <1>       mov    ch, [edi]
0000B542 EBE5                   <1>       jmp    short loc_find_free_deleted_entry_2
                                <1>
                                <1> pass_loc_check_ffde_0_err:
0000B544 38CD                   <1>       cmp    ch, cl
0000B546 741F                   <1>       je     short loc_check_ffde_attrib
                                <1>
0000B548 6643                   <1>       inc    bx
0000B54A 83C720                 <1>       add    edi, 32
0000B54D 663B1D[7F890100]       <1>       cmp    bx, [DirBuff_LastEntry]
0000B554 0F8770FFFFFF           <1>         ja      loc_ffde_stc_retn_255
0000B55A 8815[508A0100]         <1>       mov    [PreviousAttr], dl
0000B560 8A2F                   <1>       mov    ch, [edi]
0000B562 8A570B                 <1>       mov    dl, [edi+0Bh]
```

```
 982 0000B565 EBDD              <1>       jmp    short pass_loc_check_ffde_0_err
 983                            <1>
 984                            <1> loc_check_ffde_attrib:
 985 0000B567 88C6              <1>       mov    dh, al
 986 0000B569 20D6              <1>       and    dh, dl
 987 0000B56B 38F0              <1>       cmp    al, dh
 988 0000B56D 759D              <1>       jne    short loc_check_ffde_0_next
 989 0000B56F 20D4              <1>       and    ah, dl
 990 0000B571 7599              <1>       jnz    short loc_check_ffde_0_next
 991 0000B573 30C9              <1>       xor    cl, cl
 992 0000B575 EB84              <1>         jmp     loc_check_ffde_retn_2
 993                            <1>
 994                            <1> convert_file_name:
 995                            <1>       ; 06/03/2016
 996                            <1>       ; 11/02/2016
 997                            <1>       ; 07/02/2016 (TRDOS 386 = TRDOS v2.0)
 998                            <1>       ; 06/10/2009
 999                            <1>       ; 2005
1000                            <1>       ;
1001                            <1>       ; INPUT  ->
1002                            <1>       ;      ESI = Dot File Name Location
1003                            <1>       ;      EDI = Dir Entry Format File Name Location
1004                            <1>       ; OUTPUT ->
1005                            <1>       ;      EDI = Dir Entry Format File Name Location
1006                            <1>       ;      ESI = Dot File Name Location (capitalized)
1007                            <1>       ;
1008                            <1>       ; (ECX, AL will be changed)
1009                            <1>
1010 0000B577 56               <1>       push   esi
1011 0000B578 57               <1>       push   edi
1012                            <1>
1013 0000B579 B90B000000       <1>       mov    ecx, 11
1014 0000B57E B020             <1>       mov    al, 20h
1015 0000B580 F3AA             <1>       rep    stosb
1016                            <1>
1017 0000B582 8B3C24           <1>       mov    edi, [esp]
1018                            <1>
1019 0000B585 B10C             <1>       mov    cl, 12 ; file name length (max.)
1020                            <1>       ; 06/03/2016
1021                            <1>       ; Directory entry name limit (11 bytes) check for
1022                            <1>       ; 'rename_directory_entry' procedure.
1023                            <1>       ; (EDI points to Directory Entry)
1024                            <1>       ; (If the file name would not contain a dot
1025                            <1>       ; and file name length would be 12, this would cause to
1026                            <1>       ; overwrite the attributes byte of the directory entry.)
1027                            <1>       ;
1028 0000B587 B50B             <1>       mov    ch, 11 ; directory entry's name length
1029                            <1> loc_check_first_dot:
1030 0000B589 8A06             <1>       mov    al, [esi]
1031 0000B58B 3C2E             <1>       cmp    al, 2Eh
1032 0000B58D 750C             <1>       jne    short pass_check_first_dot
1033 0000B58F 8807             <1>       mov    [edi], al
1034 0000B591 47               <1>       inc    edi
1035 0000B592 46               <1>       inc    esi
1036 0000B593 FEC9             <1>       dec    cl
1037 0000B595 75F2             <1>       jnz    short loc_check_first_dot
1038                            <1>       ;;(ecx <= 12)
1039                            <1>       ;;loop loc_check_first_dot
1040 0000B597 EB30             <1>       jmp    short stop_convert_file
1041                            <1>
1042                            <1> loc_get_fchar:
1043 0000B599 8A06             <1>       mov    al, [esi]
1044                            <1> pass_check_first_dot:
1045 0000B59B 3C61             <1>       cmp    al, 61h ; 'a'
1046 0000B59D 7208             <1>       jb     short pass_name_capitalize
1047 0000B59F 3C7A             <1>       cmp    al, 7Ah ; 'z'
1048 0000B5A1 7704             <1>       ja     short pass_name_capitalize
1049 0000B5A3 24DF             <1>       and    al, 0DFh
1050 0000B5A5 8806             <1>       mov    [esi], al
1051                            <1> pass_name_capitalize:
1052 0000B5A7 3C21             <1>       cmp    al, 21h
1053 0000B5A9 721E             <1>       jb     short stop_convert_file
1054 0000B5AB 3C2E             <1>       cmp    al, 2Eh ; '.'
1055 0000B5AD 750C             <1>       jne    short pass_dot_space
1056                            <1> add_dot_space:
1057 0000B5AF 80F904           <1>       cmp    cl, 4
1058 0000B5B2 760E             <1>       jna    short inc_and_loop
1059 0000B5B4 47               <1>       inc    edi
1060 0000B5B5 FECD             <1>       dec    ch ; 06/03/2016
1061 0000B5B7 FEC9             <1>       dec    cl
1062 0000B5B9 EBF4             <1>       jmp    short add_dot_space
1063                            <1>
1064                            <1>       ;mov   al, 4
1065                            <1>       ;cmp   cl, al
1066                            <1>       ;jna   short inc_and_loop
1067                            <1>       ;sub   cl, al
1068                            <1>       ;add   edi, ecx
1069                            <1>       ;mov   cl, al
1070                            <1>       ;jmp   short inc_and_loop
1071                            <1>
1072                            <1> pass_dot_space:
1073 0000B5BB 8807             <1>       mov    [edi], al
1074                            <1> loc_after_double_dot:
1075                            <1>       ; 06/03/2016
1076 0000B5BD FECD             <1>       dec    ch ; count down for 11 bytes dir entry limit
1077 0000B5BF 740A             <1>       jz     short stop_convert_file_x
1078 0000B5C1 47               <1>       inc    edi
1079                            <1> inc_and_loop:
1080 0000B5C2 FEC9             <1>       dec    cl ; count down for 12 bytes filename limit
1081 0000B5C4 7403             <1>       jz     short stop_convert_file
1082 0000B5C6 46               <1>       inc    esi
1083                            <1>       ;;(ecx <= 12)
1084                            <1>       ;;loop loc_get_fchar
1085 0000B5C7 EBD0             <1>       jmp    short loc_get_fchar
1086                            <1>
```

```
1087                            <1> stop_convert_file:
1088                            <1>       ; 06/03/2016
1089 0000B5C9 30ED             <1>       xor    ch, ch
1090                            <1>       ; ECX < 256 ; 'find_first_file' -> xor cl, cl
1091                            <1> stop_convert_file_x:
1092 0000B5CB 5F               <1>       pop    edi
1093 0000B5CC 5E               <1>       pop    esi
1094 0000B5CD C3               <1>       retn
1095                            <1>
1096                            <1> save_longname_sub_component:
1097                            <1>       ; 13/02/2016
1098                            <1>       ; 06/02/2016 (TRDOS 386 = TRDOS v2.0)
1099                            <1>       ; 28/02/2010
1100                            <1>       ; 17/10/2009
1101                            <1>       ; INPUT ->
1102                            <1>       ;      EDI = Directory Entry
1103                            <1>       ;      // This procedure is called
1104                            <1>       ;      // from 'find_directory_entry' procedure.
1105                            <1>       ;      // If the last entry returns with
1106                            <1>       ;      // a non-zero LongnameFound value and
1107                            <1>       ;      // if LFN_CheckSum value is equal to
1108                            <1>       ;      // the next shortname checksum,
1109                            <1>       ;      // long name is valid.
1110                            <1>       ;      // If a longname is longer than 65 bytes,
1111                            <1>       ;      // it is invalid for trdos. (>45h)
1112                            <1>
1113 0000B5CE 57                <1>       push   edi
1114 0000B5CF 56               <1>       push   esi
1115                            <1>       ;push  ebx
1116                            <1>       ;push  ecx
1117                            <1>       ;push  edx
1118 0000B5D0 50               <1>       push   eax
1119                            <1>
1120 0000B5D1 29C9             <1>       sub    ecx, ecx
1121                            <1>       ;sub   eax, eax
1122 0000B5D3 B11A             <1>       mov    cl, 26
1123                            <1>
1124 0000B5D5 0FB607           <1>       movzx  eax, byte [edi] ; LDIR_Order
1125 0000B5D8 3C41             <1>       cmp    al, 41h  ; 40h (last long entry sign) + 1
1126 0000B5DA 722B             <1>       jb     short pass_pslnsc_last_long_entry
1127                            <1>
1128 0000B5DC 88C4             <1>       mov    ah, al
1129 0000B5DE 80EC40           <1>       sub    ah, 40h
1130 0000B5E1 8825[528A0100]   <1>       mov    [LFN_EntryLength], ah
1131                            <1>
1132 0000B5E7 3C45             <1>       cmp    al, 45h  ; 40h (last long entry sign) + 5
1133                            <1>              ; Max 130 byte length is usable in TRDOS
1134                            <1> ; 26*5 = 130
1135 0000B5E9 7753             <1>       ja     short loc_pslnsc_retn
1136                            <1>
1137 0000B5EB 2407             <1>       and    al, 07h ; 0Fh
1138 0000B5ED A2[518A0100]     <1>       mov    [LongNameFound], al
1139                            <1>
1140 0000B5F2 FEC8             <1>       dec    al
1141                            <1>       ;mov   cl, 26
1142 0000B5F4 F6E1             <1>       mul    cl
1143                            <1>
1144 0000B5F6 89C6             <1>       mov    esi, eax
1145 0000B5F8 01CE             <1>       add    esi, ecx
1146                            <1>              ; to make is an ASCIIZ string
1147                            <1>              ; with ax+26 bytes length
1148 0000B5FA 81C6[548A0100]   <1>       add    esi, LongFileName
1149 0000B600 66C7060000       <1>       mov    word [esi], 0
1150 0000B605 EB16             <1>       jmp    short loc_pslsc_move_ldir_name2
1151                            <1>
1152                            <1> pass_pslnsc_last_long_entry:
1153 0000B607 3C04             <1>       cmp    al, 04h
1154 0000B609 7733             <1>       ja     short loc_pslnsc_retn
1155 0000B60B FE0D[518A0100]   <1>       dec    byte [LongNameFound]
1156 0000B611 3A05[518A0100]   <1>       cmp    al, [LongNameFound]
1157 0000B617 7525             <1>       jne    short loc_pslnsc_retn
1158                            <1>
1159                            <1> loc_pslsc_move_ldir_name1:
1160 0000B619 FEC8             <1>       dec    al
1161                            <1>       ;mov   cl, 26
1162 0000B61B F6E1             <1>       mul    cl
1163                            <1>
1164                            <1> loc_pslsc_move_ldir_name2:
1165 0000B61D 8A4F0D           <1>       mov    cl, [edi+0Dh] ; long name checksum
1166 0000B620 880D[538A0100]   <1>       mov    [LFN_CheckSum], cl
1167 0000B626 89FE             <1>       mov    esi, edi ; LDIR_Order
1168 0000B628 BF[548A0100]     <1>       mov    edi, LongFileName
1169 0000B62D 01C7             <1>       add    edi, eax
1170 0000B62F 46               <1>       inc    esi
1171 0000B630 B105             <1>       mov    cl, 5 ; chars 1 to 5
1172 0000B632 F366A5           <1>       rep    movsw
1173 0000B635 83C603           <1>       add    esi, 3
1174 0000B638 A5               <1>       movsd  ; char 6 & 7
1175 0000B639 A5               <1>       movsd  ; char 8 & 9
1176 0000B63A A5               <1>       movsd  ; char 10 & 11
1177 0000B63B 46               <1>       inc    esi
1178 0000B63C 46               <1>       inc    esi
1179 0000B63D A5               <1>       movsd   ; char 12 & 13
1180                            <1>
1181                            <1> loc_pslnsc_retn:
1182 0000B63E 58               <1>       pop    eax
1183                            <1>       ;pop   edx
1184                            <1>       ;pop   ecx
1185                            <1>       ;pop   ebx
1186 0000B63F 5E               <1>       pop    esi
1187 0000B640 5F               <1>       pop    edi
1188                            <1>
1189 0000B641 C3               <1>       retn
1190                            <1>
1191                            <1> parse_path_name:
```

```
                                <1>       ; 10/02/2016
                                <1>       ; 08/02/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>       ; 10/009/2011 ('proc_parse_pathname')
                                <1>       ; 27/11/2009
                                <1>       ; 05/12/2004
                                <1>       ;
                                <1>       ; INPUT ->
                                <1>       ;     ESI = Beginning of ASCIIZ pathname string
                                <1>       ;       EDI = Destination Address
                                <1>       ;            (which is TR-DOS FindFile data buffer)
                                <1>       ; OUTPUT ->
                                <1>       ;     CF = 1 -> Error
                                <1>       ;          EAX = Error Code (AL)
                                <1>       ;
                                <1>       ; (Modified registers: eax, ecx, esi, edi)
                                <1>
                                <1>       ; Clear the pathname bytes in TR-DOS Findfile data buffer
0000B642 57                     <1>       push   edi
0000B643 B914000000             <1>       mov    ecx, 20  ; 80 bytes
0000B648 31C0                   <1>       xor    eax, eax
0000B64A F3AB                   <1>       rep    stosd
0000B64C 5F                     <1>       pop    edi
                                <1>
0000B64D 668B06                 <1>       mov    ax, [esi]
0000B650 80FC3A                 <1>       cmp    ah, ':'
0000B653 741C                   <1>       je     short loc_ppn_change_drive
0000B655 A0[56820100]           <1>       mov    al, [Current_Drv]
0000B65A EB33                   <1>       jmp    short pass_ppn_change_drive
                                <1>
                                <1> pass_ppn_cdir:
0000B65C 8B35[768B0100]         <1>       mov    esi, [First_Path_Pos]
0000B662 AC                     <1>       lodsb
                                <1> loc_ppn_get_filename:
0000B663 83C741                 <1>       add    edi, 65 ; FindFile_Name location
                                <1>       ; TRDOS Filename length must not be more than 12 bytes
                                <1>       ;mov   ecx, 12
0000B666 B10C                   <1>       mov    cl, 12
                                <1> loc_ppn_get_fnchar_next:
0000B668 AA                     <1>       stosb
0000B669 AC                     <1>       lodsb
0000B66A 3C21                   <1>       cmp    al, 21h
0000B66C 7274                   <1>       jb     short loc_ppn_clc_return
0000B66E E2F8                   <1>         loop    loc_ppn_get_fnchar_next
                                <1> loc_ppn_return:
0000B670 C3                     <1>       retn
                                <1>
                                <1> loc_ppn_change_drive:
0000B671 24DF                   <1>       and    al, 0DFh
0000B673 2C41                   <1>       sub    al, 'A'; A:
0000B675 726F                   <1>       jc     short loc_ppn_invalid_drive
0000B677 3805[22380100]         <1>       cmp    [Last_DOS_DiskNo], al
0000B67D 7267                   <1>       jb     short loc_ppn_invalid_drive
                                <1>
0000B67F 46                     <1>       inc    esi
0000B680 46                     <1>       inc    esi
0000B681 8A26                   <1>       mov    ah, [esi]
0000B683 80FC21                 <1>       cmp    ah, 21h
0000B686 7307                   <1>       jnb    short pass_ppn_change_drive
                                <1>
                                <1> loc_ppn_cmd_failed:
                                <1>       ; File or directory name is not existing
0000B688 8807                   <1>       mov    [edi], al ; Drv
0000B68A 66B80100               <1>       mov    ax, 1 ; eax = 1
                                <1>       ; TR-DOS Error Code 01h = Bad Command Argument
                                <1>       ; MS-DOS Error Code 01h : Invalid Function Number
                                <1>       ;stc
                                <1>       ; (MainProg ErrMsg: "Bad command or file name!")
0000B68E C3                     <1>       retn
                                <1>
                                <1> pass_ppn_change_drive:
0000B68F 8935[768B0100]         <1>       mov    [First_Path_Pos], esi
0000B695 C705[7A8B0100]0000-    <1>       mov    dword [Last_Slash_Pos], 0
0000B69D 0000                   <1>
0000B69F AA                     <1>       stosb
0000B6A0 8A06                   <1>       mov    al, [esi]
                                <1> loc_scan_ppn_dslash:
0000B6A2 3C2F                   <1>       cmp    al, '/'
0000B6A4 7506                   <1>       jne    short loc_scan_next_slash_pos
0000B6A6 8935[7A8B0100]         <1>       mov    [Last_Slash_Pos], esi
                                <1> loc_scan_next_slash_pos:
0000B6AC 46                     <1>       inc    esi
0000B6AD 8A06                   <1>       mov    al, [esi]
0000B6AF 3C20                   <1>       cmp    al, 20h
0000B6B1 77EF                   <1>       ja     short loc_scan_ppn_dslash
0000B6B3 833D[7A8B0100]00       <1>       cmp    dword [Last_Slash_Pos], 0
0000B6BA 76A0                   <1>       jna    short pass_ppn_cdir
                                <1>
0000B6BC 8B0D[7A8B0100]         <1>       mov    ecx, [Last_Slash_Pos]
0000B6C2 8B35[768B0100]         <1>       mov    esi, [First_Path_Pos]
0000B6C8 29F1                   <1>       sub    ecx, esi
0000B6CA 41                     <1>       inc    ecx
                                <1>       ;cmp   ecx, 64
0000B6CB 80F940                 <1>       cmp    cl, 64
0000B6CE 7715                   <1>       ja     short loc_ppn_invalid_drive_stc
                                <1>
0000B6D0 89F8                   <1>       mov    eax, edi ; Dest Dir String Location (65 byte)
0000B6D2 F3A4                   <1>       rep    movsb
                                <1>       ;mov   [edi], cl ; 0, End of Dir String
0000B6D4 8B35[7A8B0100]         <1>       mov    esi, [Last_Slash_Pos]
0000B6DA 46                     <1>       inc    esi
0000B6DB 89C7                   <1>       mov    edi, eax
0000B6DD AC                     <1>       lodsb
0000B6DE 3C21                   <1>       cmp    al, 21h
0000B6E0 7381                   <1>       jnb    short loc_ppn_get_filename
                                <1> loc_ppn_clc_return:
```

```
1296                                    <1>       ;clc
1297 0000B6E2 31C0                      <1>       xor    eax, eax
1298 0000B6E4 C3                        <1>       retn
1299                                    <1>
1300                                    <1> loc_ppn_invalid_drive_stc:
1301 0000B6E5 F5                        <1>       cmc     ; stc
1302                                    <1> loc_ppn_invalid_drive:
1303                                    <1>       ; cf = 1
1304                                    <1>       ; The Drive Letter/Char < "A" or > "Z"
1305 0000B6E6 66B80F00                  <1>       mov    ax, 0Fh
1306                                    <1>       ; MS-DOS Error Code 0Fh = Disk Drive Invalid
1307                                    <1>       ; (MainProg ErrMsg: "Drive not ready or read error!")
1308 0000B6EA C3                        <1>       retn
1309                                    <1>
1310                                    <1> find_longname:
1311                                    <1>       ; 13/02/2016 (TRDOS 386 = TRDOS v2.0)
1312                                    <1>       ; 24/01/2010 (DIR.ASM, 'proc_find_longname')
1313                                    <1>       ; 17/10/2009
1314                                    <1>
1315                                    <1>       ; INPUT ->
1316                                    <1>       ;      ESI = DOS short file name address
1317                                    <1>       ;      for example: "filename.ext"
1318                                    <1>       ;
1319                                    <1>       ; OUTPUT ->
1320                                    <1>       ;      ESI = ASCIIZ longname address (cf = 0)
1321                                    <1>       ;      cf = 1 -> error number returns in EAX (AL)
1322                                    <1>       ;      AL = 0 & CF=1 -> longname not found
1323                                    <1>       ;           the file/directory has no longname
1324                                    <1>       ;      cf = 0 -> AL = FAT Type
1325                                    <1>
1326                                    <1>       ; 17/10/2009
1327                                    <1>       ; ASCIIZ string will be returned
1328                                    <1>       ; as LongFileName
1329                                    <1>       ; clearing/reset is not needed
1330                                    <1>       ;mov   ecx, 33
1331                                    <1>       ;mov   edi, LongFileName
1332                                    <1>       ;sub   ax, ax ; 0
1333                                    <1>       ;rep   stosw
1334                                    <1>
1335                                    <1>       ;mov   byte [LongNameFound], 0
1336                                    <1>
1337                                    <1>       ; ESI = ASCIIZ file/directory name address
1338                                    <1>       ;   AL = Attributes AND mask
1339                                    <1>       ;      (Result of AND must be equal to AL)
1340                                    <1>       ;   AH = Negative attributes mask
1341                                    <1>       ;      (Result of AND must be ZERO)
1342 0000B6EB 66B80008                  <1>       mov    ax, 0800h
1343                                    <1>              ; it must not be volume name or longname
1344 0000B6EF E87DDDFFFF                <1>       call   find_first_file
1345 0000B6F4 7216                      <1>       jc     short loc_fln_retn
1346                                    <1>
1347                                    <1> loc_fln_check_FAT_Type:
1348 0000B6F6 803D[55820100]01          <1>       cmp    byte [Current_FATType], 1
1349 0000B6FD 7306                      <1>       jnb    short loc_fln_check_longname_yes_sign
1350                                    <1>
1351 0000B6FF E839000000                <1>       call   get_fs_longname
1352 0000B704 C3                        <1>       retn
1353                                    <1>
1354                                    <1> loc_fln_check_longname_yes_sign:
1355 0000B705 08FF                      <1>       or     bh, bh
1356 0000B707 7504                      <1>       jnz    short loc_fln_check_longnamefound_number
1357                                    <1> loc_fln_longname_not_found_retn:
1358 0000B709 31C0                      <1>       xor    eax, eax
1359                                    <1>       ; cf = 1 & al = 0 -> longname not found
1360 0000B70B F9                        <1>       stc
1361                                    <1> loc_fln_retn:
1362 0000B70C C3                        <1>       retn
1363                                    <1>
1364                                    <1> loc_fln_check_longnamefound_number:
1365                                    <1>       ; 'LongNameFound' is set by
1366                                    <1>         ; by 'save_longname_sub_component'
1367                                    <1>       ; which is called from
1368                                    <1>       ; 'find_directory_entry'
1369                                    <1>       ; which is called from
1370                                    <1>       ; 'find_first_file'
1371                                    <1>       ; It must 1 if the longname is valid
1372 0000B70D 803D[518A0100]01          <1>         cmp     byte [LongNameFound], 1
1373 0000B714 75F3                      <1>       jne    short loc_fln_longname_not_found_retn
1374                                    <1>
1375                                    <1> loc_fln_calculate_checksum:
1376 0000B716 E813000000                <1>       call   calculate_checksum
1377                                    <1>       ; AL = shortname checksum
1378                                    <1>
1379                                    <1> loc_fln_longname_validation:
1380                                    <1>       ; 'LFN_CheckSum' has been set already
1381                                    <1>       ; by 'save_longname_sub_component'
1382                                    <1>       ; which is called from
1383                                    <1>       ; 'find_directory_entry'
1384                                    <1>       ; which is called from
1385                                    <1>       ; 'find_first_file'
1386 0000B71B 3805[538A0100]            <1>       cmp    [LFN_CheckSum], al
1387 0000B721 75E6                      <1>       jne    short loc_fln_longname_not_found_retn
1388                                    <1>
1389 0000B723 BE[548A0100]              <1>       mov    esi, LongFileName
1390 0000B728 A0[55820100]              <1>       mov    al, [Current_FATType]
1391 0000B72D C3                        <1>       retn
1392                                    <1>
1393                                    <1> calculate_checksum:
1394                                    <1>       ; 13/02/2016 (TRDOS 386 = TRDOS v2.0)
1395                                    <1>       ; 17/10/2009 (DIR.ASM, 'proc_calculate_checksum')
1396                                    <1>         ;
1397                                    <1>       ; INPUT ->
1398                                    <1>       ;      ESI = 11 byte DOS File Name location
1399                                    <1>       ;      (in DOS Directory Entry Format)
1400                                    <1>       ; OUTPUT ->
```

```
1401                                <1>       ;       AL = 8 bit checksum (CRC) value
1402                                <1>       ;
1403                                <1>       ; (Modified registers: EAX, ECX, ESI)
1404                                <1>
1405                                <1>       ; Erdogan Tan [ 17-10-2009 ]
1406                                <1>       ;  'ror al, 1' instruction
1407                                <1>
1408                                <1>       ; Erdogan Tan [ 20-06-2004 ]
1409                                <1>       ; This 8086 assembly code is an original code
1410                                <1>       ; which is adapted from C code in
1411                                <1>       ; Microsoft FAT32 File System Specification
1412                                <1>       ; Version 1.03, December 6, 2000
1413                                <1>       ; Page 28
1414                                <1>
1415 0000B72E 30C0                  <1>       xor    al, al
1416 0000B730 B90B000000            <1>       mov    ecx, 11
1417                                <1> loc_next_sum:
1418                                <1>       ;xor   ah, ah
1419                                <1>       ;test  al, 1
1420                                <1>       ;jz    short pass_ah_80h
1421                                <1>       ;mov   ah, 80h
1422                                <1> ;pass_ah_80h:
1423                                <1>       ;shr   al, 1
1424 0000B735 D0C8                  <1>       ror    al, 1 ; 17/10/2009
1425 0000B737 0206                  <1>       add    al, [esi]
1426 0000B739 46                    <1>       inc    esi
1427                                <1>       ;add   al, ah
1428 0000B73A E2F9                  <1>       loop   loc_next_sum
1429 0000B73C C3                    <1>       retn
1430                                <1>
1431                                <1> get_fs_longname:
1432                                <1>       ; temporary (13/02/2016)
1433 0000B73D 31C0                  <1>       xor eax, eax
1434 0000B73F F9                    <1>       stc
1435 0000B740 C3                    <1>       retn
1436                                <1>
1437                                <1> make_sub_directory:
1438                                <1>       ; 16/10/2016
1439                                <1>       ; 02/03/2016, 03/03/2016
1440                                <1>       ; 26/02/2016, 27/02/2016
1441                                <1>       ; 21/02/2016 (TRDOS 386 = TRDOS v2.0)
1442                                <1>       ; 01/08/2011 (DIR.ASM, 'proc_make_directory')
1443                                <1>       ; 10/07/2010
1444                                <1>       ; INPUT ->
1445                                <1>       ;      ESI = ASCIIZ Directory Name
1446                                <1>       ;      CL = Directory Attributes
1447                                <1>       ; OUTPUT ->
1448                                <1>       ;      EAX = New sub dir's first cluster
1449                                <1>       ;      ESI = Logical Dos Drv Descr. Table Addr.
1450                                <1>       ;      CF = 1 -> error code in AL (EAX)
1451                                <1>
1452                                <1>       ;test  cl, 10h  ; directory
1453                                <1>       ;jz    short loc_make_directory_access_denied
1454                                <1>       ;test  cl, 08h ; volume name
1455                                <1>       ;jnz   short loc_make_directory_access_denied
1456                                <1>
1457 0000B741 80E107                <1>       and    cl, 07h
1458 0000B744 880D[D08B0100]        <1>       mov    byte [mkdir_attrib], cl
1459                                <1>
1460 0000B74A 56                    <1>       push   esi
1461 0000B74B 31DB                  <1>       xor    ebx, ebx
1462 0000B74D 8A3D[56820100]        <1>       mov    bh, [Current_Drv]
1463 0000B753 BE00010900            <1>       mov    esi, Logical_DOSDisks
1464 0000B758 01DE                  <1>       add    esi, ebx
1465 0000B75A 5B                    <1>       pop    ebx
1466                                <1>
1467                                <1>       ; 10/07/2010 -> 1st writable disk check for trdos
1468                                <1>       ; LD_DiskType = 0 for write protection (read only)
1469 0000B75B 807E0101              <1>       cmp    byte [esi+LD_DiskType], 1 ; 0 = Invalid
1470 0000B75F 730B                  <1>       jnb    short loc_mkdir_check_file_sytem
1471                                <1>       ; 16/10/2016 (13h -> 30)
1472 0000B761 B81E000000            <1>       mov    eax, 30 ; 'Disk write-protected' error
1473 0000B766 BA00000000            <1>       mov    edx, 0
1474                                <1>       ; err retn: EDX = 0, EBX = Dir name offset
1475                                <1>       ;ESI = Logical DOS drive description table address
1476 0000B76B C3                    <1>       retn
1477                                <1>
1478                                <1> ;loc_make_directory_access_denied:
1479                                <1>       ;mov   ax, 05h ; access denied (invalid attributes input)
1480                                <1>       ;stc
1481                                <1>       ;retn
1482                                <1>
1483                                <1> loc_mkdir_check_file_sytem:
1484 0000B76C 807E0301              <1>       cmp    byte [esi+LD_FATType], 1
1485 0000B770 730B                  <1>       jnb    short loc_mkdir_check_free_sectors
1486                                <1>
1487                                <1> loc_make_fs_directory:
1488 0000B772 A1[50820100]          <1>       mov    eax, [Current_Dir_FCluster]
1489                                <1>       ; EAX = Parent directory DDT Address
1490                                <1>       ; ESI = Logical DOS Drive DT Address
1491                                <1>       ; EBX = Directory name offset (as ASCIIZ name)
1492 0000B777 E8D5150000            <1>       call   make_fs_directory
1493 0000B77C C3                    <1>       retn
1494                                <1>
1495                                <1> loc_mkdir_check_free_sectors:
1496 0000B77D 0FB64613              <1>         movzx   eax, byte [esi+LD_BPB+SecPerClust]
1497 0000B781 8B4E74                <1>       mov    ecx, [esi+LD_FreeSectors]
1498 0000B784 39C1                  <1>       cmp    ecx, eax
1499 0000B786 7255                  <1>       jb     short loc_mkdir_insufficient_disk_space
1500                                <1>
1501                                <1> loc_make_fat_directory:
1502 0000B788 891D[C08B0100]        <1>       mov    [mkdir_DirName_Offset], ebx
1503 0000B78E 890D[CC8B0100]        <1>       mov    [mkdir_FreeSectors], ecx
1504                                <1>
1505                                <1>       ;mov   al, [esi+LD_BPB+SecPerClust]
```

```
1506 0000B794 A2[D28B0100]        <1>       mov   byte [mkdir_SecPerClust], al
1507                              <1>
1508                              <1> loc_mkdir_gffc_1:
1509 0000B799 E80F180000          <1>       call  get_first_free_cluster
1510 0000B79E 722A                <1>       jc    short loc_mkdir_gffc_retn
1511                              <1>
1512                              <1> ;loc_mkdir_gffc_1_cont:
1513                              <1>       ;cmp  eax, 2
1514                              <1>       ;jb   short loc_mkdir_gffc_insufficient_disk_space
1515                              <1>
1516                              <1> ;loc_mkdir_gffc_1_save_fcluster:
1517 0000B7A0 A3[C48B0100]        <1>       mov   [mkdir_FFCluster], eax
1518                              <1>
1519                              <1> loc_mkdir_locate_ffe:
1520                              <1>       ; Current directory fcluster <> Directory buffer cluster
1521                              <1>       ; Current directory will be reloaded by
1522                              <1>       ; 'locate_current_dir_file' procedure
1523                              <1>       ;
1524                              <1>       ; ESI = Logical DOS Drive Description Table Address
1525                              <1>       ;push  esi ; 27/02/2016
1526 0000B7A5 31C0                <1>       xor   eax, eax
1527 0000B7A7 89C1                <1>         mov ecx, eax
1528 0000B7A9 6649                <1>       dec   cx ; FFFFh
1529                              <1>       ; CX = FFFFh -> find first deleted or free entry
1530                              <1>       ; ESI would be ASCIIZ filename address if the call
1531                              <1>       ; would not be for first free or deleted dir entry
1532 0000B7AB E8D0FAFFFF          <1>       call  locate_current_dir_file
1533 0000B7B0 734C                <1>       jnc   short loc_mkdir_set_ff_dir_entry_1
1534                              <1>       ;pop   esi
1535                              <1>       ; ESI = Logical DOS Drive Description Table Address
1536 0000B7B2 83F802              <1>       cmp   eax, 2  ; cmp al, 2 ; File/Dir not found !
1537 0000B7B5 752B                <1>       jne   short loc_mkdir_stc_return
1538                              <1>
1539                              <1> loc_mkdir_add_new_cluster:
1540 0000B7B7 3805[55820100]      <1>       cmp   byte [Current_FATType], al ; 2
1541                              <1>       ;cmp  byte ptr [esi+LD_FATType], 2
1542 0000B7BD 770C                <1>       ja    short loc_mkdir_add_new_cluster_check_fsc
1543 0000B7BF 803D[54820100]01    <1>       cmp   byte [Current_Dir_Level], 1
1544                              <1>       ;cmp  byte [esi+LD_CDirLevel], 1
1545 0000B7C6 7303                <1>       jnb   short loc_mkdir_add_new_cluster_check_fsc
1546                              <1>
1547 0000B7C8 B00C                <1>       mov   al, 12 ; No more files
1548                              <1> loc_mkdir_gffc_retn:
1549 0000B7CA C3                  <1>       retn
1550                              <1>
1551                              <1> loc_mkdir_add_new_cluster_check_fsc:
1552 0000B7CB 8B0D[CC8B0100]      <1>       mov   ecx, [mkdir_FreeSectors]
1553                              <1>       ;movzx eax, byte [mkdir_SecPerClust]
1554 0000B7D1 A0[D28B0100]        <1>       mov   al, [mkdir_SecPerClust]
1555 0000B7D6 66D1E0              <1>       shl   ax, 1 ; AX = 2 * AX
1556 0000B7D9 39C1                <1>       cmp   ecx, eax
1557 0000B7DB 7350                <1>       jnb   short loc_mkdir_add_new_subdir_cluster
1558                              <1>
1559                              <1> loc_mkdir_insufficient_disk_space:
1560                              <1>       ;mov  edx, ecx
1561                              <1> ;loc_mkdir_gffc_insufficient_disk_space:
1562 0000B7DD 66B82700              <1>       mov   ax, 27h ; MSDOS err => insufficient disk space
1563                              <1>       ; err retn: EDX = Free sectors, EBX = Dir name offset
1564                              <1>         ; ESI -> Dos drive description table address
1565                              <1>       ;; ecx = edx
1566                              <1>       ;
1567 0000B7E1 C3                  <1>       retn
1568                              <1>
1569                              <1> loc_mkdir_stc_return:
1570 0000B7E2 F9                  <1>       stc
1571 0000B7E3 C3                  <1>       retn
1572                              <1>
1573                              <1> loc_mkdir_gffc_2:
1574 0000B7E4 E8C4170000          <1>       call  get_first_free_cluster
1575 0000B7E9 72DF                <1>       jc    short loc_mkdir_gffc_retn
1576                              <1>
1577                              <1> ;loc_mkdir_gffc_1_cont:
1578                              <1>       ;cmp  eax, 2
1579                              <1>       ;jb   short loc_mkdir_gffc_insufficient_disk_space
1580                              <1>
1581                              <1> ;loc_mkdir_gffc_2_save_fcluster:
1582 0000B7EB A3[C48B0100]        <1>       mov   [mkdir_FFCluster], eax
1583                              <1>
1584 0000B7F0 A1[C88B0100]        <1>       mov   eax, [mkdir_LastDirCluster]
1585                              <1>
1586 0000B7F5 E842170000          <1>       call  load_FAT_sub_directory
1587 0000B7FA 72CE                <1>       jc    short loc_mkdir_gffc_retn
1588                              <1>
1589 0000B7FC 31FF                <1>       xor   edi, edi
1590                              <1> loc_mkdir_set_ff_dir_entry_1:
1591                              <1>       ; 27/02/2016
1592 0000B7FE 56                  <1>       push  esi ; Logical DOS Drv Desc. Tbl. address
1593                              <1>       ; EDI = Directory Entry Address
1594 0000B7FF 8B35[C08B0100]      <1>       mov   esi, [mkdir_DirName_Offset]
1595 0000B805 A1[C48B0100]        <1>       mov   eax, [mkdir_FFCluster]
1596                              <1>
1597 0000B80A 66B91000            <1>       mov   cx, 10h      ; CL = Directory attribute
1598                              <1>                          ; CH = 0 -> File size is 0
1599 0000B80E 0A0D[D08B0100]      <1>       or    cl, [mkdir_attrib] ; S, H, R
1600 0000B814 E8B0010000          <1>       call  make_directory_entry
1601                              <1>
1602 0000B819 5E                  <1>       pop   esi
1603                              <1>
1604 0000B81A C605[7C890100]02    <1>       mov   byte [DirBuff_ValidData], 2
1605 0000B821 E880020000          <1>       call  save_directory_buffer
1606 0000B826 0F83DA000000        <1>         jnc     loc_mkdir_set_ff_dir_entry_2
1607                              <1>
1608                              <1> loc_mkdir_return:
1609 0000B82C C3                  <1>       retn
1610                              <1>
```

```
1611                                <1> loc_mkdir_add_new_subdir_cluster:
1612 0000B82D 8B15[81890100]       <1>         mov     edx, [DirBuff_Cluster]
1613 0000B833 8915[C88B0100]       <1>         mov     [mkdir_LastDirCluster], edx
1614                                <1>
1615 0000B839 A1[C48B0100]         <1>         mov     eax, [mkdir_FFCluster]
1616 0000B83E E8F9160000           <1>         call    load_FAT_sub_directory
1617 0000B843 72E7                 <1>         jc      short loc_mkdir_return
1618                                <1>         ; eax = 0
1619                                <1>         ; ecx =  directory buffer sector count (<= 128)
1620                                <1>
1621                                <1> pass_mkdir_add_new_subdir_cluster:
1622 0000B845 29FF                 <1>         sub     edi, edi ; 0
1623                                <1>         ;mov    al, 128 ; double word
1624                                <1>         ;mul    ecx ; ecx =  directory buffer sector count
1625                                <1>         ;mov    ecx, eax
1626                                <1>         ;shl    cx, 7 ; 128 * sector count
1627 0000B847 668B4611             <1>         mov     ax, [esi+LD_BPB+BytesPerSec] ; 512
1628 0000B84B 66C1E802             <1>         shr     ax, 2 ; 'byte count / 4' for 'stosd'
1629 0000B84F 66F7E1               <1>         mul     cx ; max = 128*(512/4) -> 16384 (stosd)
1630 0000B852 6689C1               <1>         mov     cx, ax
1631 0000B855 6629C0               <1>         sub     ax, ax ; 0
1632 0000B858 F3AB                 <1>         rep     stosd ; clear directory buffer
1633                                <1>
1634 0000B85A C605[7C890100]02     <1>         mov     byte [DirBuff_ValidData], 2
1635 0000B861 E840020000           <1>         call    save_directory_buffer
1636 0000B866 72C4                 <1>         jc      short loc_mkdir_return
1637                                <1>
1638                                <1> loc_mkdir_save_added_cluster:
1639 0000B868 A1[C88B0100]         <1>         mov     eax, [mkdir_LastDirCluster]
1640 0000B86D 8B0D[C48B0100]       <1>         mov     ecx, [mkdir_FFCluster]
1641                                <1>         ; 01/03/2016
1642 0000B873 31D2                 <1>         xor     edx, edx
1643 0000B875 8915[72890100]       <1>         mov     [FAT_ClusterCounter], edx ; 0 ; reset
1644 0000B87B E800180000           <1>         call    update_cluster
1645 0000B880 7304                 <1>         jnc     short loc_mkdir_save_fat_buffer_0
1646 0000B882 09C0                 <1>         or      eax, eax ; EAX = 0 -> cluster value is 0 or eocc
1647 0000B884 7518                 <1>         jnz     short loc_mkdir_save_fat_buffer_stc_retn
1648                                <1>
1649                                <1> loc_mkdir_save_fat_buffer_0:
1650 0000B886 A1[C48B0100]         <1>         mov     eax, [mkdir_FFCluster]
1651 0000B88B A3[C88B0100]         <1>         mov     [mkdir_LastDirCluster], eax
1652                                <1>
1653 0000B890 31C9                 <1>         xor     ecx, ecx
1654 0000B892 49                   <1>         dec     ecx ; FFFFFFFFh
1655                                <1>         ; ESI = Logical DOS Drive Description Table address
1656 0000B893 E8E8170000           <1>         call    update_cluster
1657 0000B898 731A                 <1>         jnc     short loc_mkdir_save_fat_buffer_1
1658 0000B89A 09C0                 <1>         or      eax, eax
1659 0000B89C 7416                 <1>         jz      short loc_mkdir_save_fat_buffer_1
1660                                <1>
1661                                <1> loc_mkdir_save_fat_buffer_stc_retn:
1662                                <1>         ; 01/03/2016
1663 0000B89E 803D[72890100]01     <1>         cmp     byte [FAT_ClusterCounter], 1
1664 0000B8A5 720C                 <1>         jb      short loc_mkdir_save_fat_buffer_retn
1665                                <1>
1666 0000B8A7 66BB00FF             <1>         mov     bx, 0FF00h ; recalculate free space (BL = 0)
1667                                <1>                            ; (BH = FFh -> Use ESI as Drv Param. Tbl.)
1668 0000B8AB 50                   <1>         push    eax
1669 0000B8AC E8211B0000           <1>         call    calculate_fat_freespace
1670 0000B8B1 58                   <1>         pop     eax
1671 0000B8B2 F9                   <1>         stc
1672                                <1> loc_mkdir_save_fat_buffer_retn:
1673 0000B8B3 C3                   <1>         retn
1674                                <1>
1675                                <1> loc_mkdir_save_fat_buffer_1:
1676                                <1>         ; byte [FAT_BuffValidData] = 2
1677 0000B8B4 E8841A0000           <1>         call    save_fat_buffer
1678 0000B8B9 72E3                 <1>         jc      short loc_mkdir_save_fat_buffer_stc_retn
1679                                <1>
1680                                <1>         ; 01/03/2016
1681 0000B8BB 803D[72890100]01     <1>         cmp     byte [FAT_ClusterCounter], 1
1682 0000B8C2 721B                 <1>         jb      short loc_mkdir_save_fat_buffer_2
1683                                <1>
1684                                <1>         ; ESI = Logical DOS Drive Description Table address
1685 0000B8C4 A1[72890100]         <1>         mov     eax, [FAT_ClusterCounter]
1686 0000B8C9 66BB01FF             <1>         mov     bx, 0FF01h ; add free clusters
1687 0000B8CD E8001B0000           <1>         call    calculate_fat_freespace
1688                                <1>
1689                                <1>         ;inc    eax ; 0FFFFFFFFh -> 0 ; recalculation is needed!
1690                                <1>         ;jnz    short loc_mkdir_save_fat_buffer_2
1691                                <1>
1692                                <1>         ; ecx > 0 -> Recalculation is needed
1693 0000B8D2 09C9                 <1>         or      ecx, ecx
1694 0000B8D4 7409                 <1>         jz      short loc_mkdir_save_fat_buffer_2
1695                                <1>
1696 0000B8D6 66BB00FF             <1>         mov     bx, 0FF00h ; ; recalculate free space
1697 0000B8DA E8F31A0000           <1>         call    calculate_fat_freespace
1698                                <1>
1699                                <1> loc_mkdir_save_fat_buffer_2:
1700 0000B8DF C605[D38B0100]01     <1>         mov     byte [mkdir_add_new_cluster], 1
1701 0000B8E6 E9C4000000           <1>         jmp     loc_mkdir_upd_parent_dir_lmdt
1702                                <1>
1703                                <1> loc_mkdir_update_sub_dir_cluster:
1704 0000B8EB A1[C48B0100]         <1>         mov     eax, [mkdir_FFCluster]
1705 0000B8F0 29C9                 <1>         sub     ecx, ecx ; 0
1706                                <1>         ; 01/03/2016
1707 0000B8F2 890D[72890100]       <1>         mov     [FAT_ClusterCounter], ecx ; 0 ; Reset
1708 0000B8F8 49                   <1>         dec     ecx ; 0FFFFFFFFh
1709                                <1>
1710                                <1>         ; ESI = Logical DOS Drive Desciption Table address
1711 0000B8F9 E882170000           <1>         call    update_cluster
1712 0000B8FE 7379                 <1>         jnc     short loc_mkdir_save_fat_buffer_3
1713 0000B900 09C0                 <1>         or      eax, eax ; EAX = 0 -> cluster value is 0 or eocc
1714 0000B902 7475                 <1>         jz      short loc_mkdir_save_fat_buffer_3
1715                                <1>         ; 01/03/2016
```

```
1716 0000B904 EB98                <1>       jmp    short loc_mkdir_save_fat_buffer_stc_retn
1717                              <1>
1718                              <1> loc_mkdir_set_ff_dir_entry_2:
1719                              <1>       ; ESI = Logical DOS Drive Description Table address
1720 0000B906 A1[C48B0100]        <1>       mov    eax, [mkdir_FFCluster]
1721                              <1>       ; Load disk sectors as a directory cluster
1722 0000B90B E82C160000          <1>       call   load_FAT_sub_directory
1723 0000B910 7266                <1>       jc     short retn_make_fat_directory
1724                              <1>
1725                              <1>       ; eax = 0
1726                              <1>       ; ecx =  directory buffer sector count (<= 128)
1727                              <1>
1728 0000B912 BF40000800          <1>       mov    edi, Directory_Buffer + 64 ; 26/02/2016
1729                              <1>
1730                              <1>       ; 02/03/2016
1731 0000B917 668B4611            <1>       mov    ax, [esi+LD_BPB+BytesPerSec] ; 512
1732 0000B91B 66C1E802            <1>       shr    ax, 2 ; 'byte count / 4' for 'stosd'
1733 0000B91F F7E1                <1>       mul    ecx
1734 0000B921 89C1                <1>       mov    ecx, eax
1735 0000B923 6629C0              <1>       sub    ax, ax
1736 0000B926 F3AB                <1>       rep    stosd
1737                              <1>
1738                              <1>       ;;mov  al, 128 ; double word
1739                              <1>       ;;mul  ecx ; ecx =  directory buffer sector count
1740                              <1>       ;;mov  ecx, eax
1741                              <1>       ;shl   cx, 7 ; 128 * sector count
1742                              <1>       ;;sub  eax, eax
1743                              <1>       ;;sub  al, al ; 0
1744                              <1>       ;rep   stosd ; clear directory buffer
1745                              <1>
1746 0000B928 BF00000800          <1>       mov    edi, Directory_Buffer ; 26/02/2016
1747                              <1>
1748 0000B92D 56                  <1>       push   esi
1749                              <1>
1750 0000B92E BE[D48B0100]        <1>       mov    esi, mkdir_Name
1751 0000B933 66C7062E00          <1>       mov    word [esi], 2Eh ; db '.', '0'
1752                              <1>
1753 0000B938 A1[C48B0100]        <1>       mov    eax, [mkdir_FFCluster]
1754 0000B93D 66B91000            <1>       mov    cx, 10h ; CL = Directory attribute
1755                              <1>                      ; CH = 0 -> File size is 0
1756 0000B941 E883000000          <1>       call   make_directory_entry
1757                              <1>
1758 0000B946 BF20000800          <1>       mov    edi, Directory_Buffer + 32 ; 26/02/2016
1759                              <1>
1760                              <1>       ; 03/03/2016
1761                              <1>       ; Following modification has been done according to
1762                              <1>       ; 'Microsoft Extensible Firmware Initiative
1763                              <1>       ; FAT32 File System Specification' document,
1764                              <1>       ; 'FAT: General Overview of On-Disk Format—Page 25'.
1765                              <1>       ; "Finally, you set DIR_FstClusLO and DIR_FstClusHI
1766                              <1>       ; for the dotdot entry (the second entry) to the
1767                              <1>       ; first cluster number of the directory in which you
1768                              <1>       ; just created the directory (value is 0 if this directory
1769                              <1>       ; is the root directory even for FAT32 volumes)."
1770                              <1>       ; (Correctness of this modification has been verified
1771                              <1>       ;  by using Windows 98 'scandisk.exe'.)
1772                              <1>
1773 0000B94B 29C0                <1>       sub    eax, eax
1774 0000B94D 3805[54820100]      <1>       cmp    byte [Current_Dir_Level], al ; 0
1775 0000B953 7605                <1>       jna    short loc_mkdir_set_ff_dir_entry_3
1776 0000B955 A1[50820100]        <1>       mov    eax, [Current_Dir_FCluster] ; parent dir
1777                              <1> loc_mkdir_set_ff_dir_entry_3:
1778 0000B95A 66C746012E00        <1>       mov    word [esi+1], 2Eh ; db '.', '0'
1779                              <1>
1780                              <1>       ;mov   cx, 10h
1781 0000B960 E864000000          <1>       call   make_directory_entry
1782                              <1>
1783 0000B965 5E                  <1>       pop    esi
1784                              <1>
1785 0000B966 C605[7C890100]02    <1>       mov    byte [DirBuff_ValidData], 2
1786 0000B96D E834010000          <1>       call   save_directory_buffer
1787 0000B972 0F8373FFFFFF        <1>         jnc     loc_mkdir_update_sub_dir_cluster
1788                              <1>
1789                              <1> retn_make_fat_directory:
1790 0000B978 C3                  <1>       retn
1791                              <1>
1792                              <1> loc_mkdir_save_fat_buffer_3:
1793                              <1>       ; 01/03/2016
1794                              <1>       ; byte [FAT_BuffValidData] = 2
1795 0000B979 E8BF190000          <1>       call   save_fat_buffer
1796 0000B97E 0F821AFFFFFF        <1>         jc      loc_mkdir_save_fat_buffer_stc_retn
1797                              <1>
1798 0000B984 803D[72890100]01    <1>       cmp    byte [FAT_ClusterCounter], 1
1799 0000B98B 721B                <1>       jb     short loc_mkdir_save_fat_buffer_4
1800                              <1>
1801                              <1>       ; ESI = Logical DOS Drive Description Table address
1802 0000B98D A1[72890100]        <1>       mov    eax, [FAT_ClusterCounter]
1803 0000B992 66BB01FF            <1>       mov    bx, 0FF01h ; add free clusters
1804 0000B996 E8371A0000          <1>       call   calculate_fat_freespace
1805                              <1>
1806                              <1>       ;inc   eax ; 0FFFFFFFFh -> 0 ; recalculation is needed!
1807                              <1>         ;jnz    short loc_mkdir_save_fat_buffer_4
1808                              <1>
1809                              <1>       ; ecx > 0 -> Recalculation is needed
1810 0000B99B 09C9                <1>       or     ecx, ecx
1811 0000B99D 7409                <1>         jz      short loc_mkdir_save_fat_buffer_4
1812                              <1>
1813 0000B99F 66BB00FF            <1>       mov    bx, 0FF00h ; recalculate free space
1814 0000B9A3 E82A1A0000          <1>       call   calculate_fat_freespace
1815                              <1>
1816                              <1> loc_mkdir_save_fat_buffer_4:
1817 0000B9A8 C605[D38B0100]00    <1>       mov    byte [mkdir_add_new_cluster], 0
1818                              <1>
1819                              <1> loc_mkdir_upd_parent_dir_lmdt:
1820 0000B9AF E88D010000          <1>       call   update_parent_dir_lmdt
```

```
                                <1>
                                <1>       ; 01/03/2016
0000B9B4 803D[D38B0100]00       <1>       cmp    byte [mkdir_add_new_cluster], 0
0000B9BB 0F8723FEFFFF           <1>         ja      loc_mkdir_gffc_2
                                <1>
                                <1> loc_mkdir_retn_new_dir_cluster:
0000B9C1 A1[C48B0100]           <1>       mov    eax, [mkdir_FFCluster]
0000B9C6 31D2                   <1>       xor    edx, edx
                                <1> loc_mkdir_retn:
0000B9C8 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>
0000B9C9 51                     <1>       push   ecx
                                <1>
0000B9CA 884F0B                 <1>       mov    [edi+11], cl ; Attributes
0000B9CD 6689471A               <1>       mov    [edi+26], ax ; FClusterLw, 26
0000B9D1 C1E810                 <1>       shr    eax, 16
0000B9D4 66894714               <1>       mov    [edi+20], ax ; FClusterHw, 20
0000B9D8 6631C0                 <1>       xor    ax, ax
0000B9DB 6689470C               <1>       mov    [edi+12], ax ; NTReserved, 12
                                <1>                           ; CrtTimeTenth, 13
0000B9DF 08ED                   <1>       or     ch, ch
0000B9E1 7402                   <1>       jz     short loc_make_direntry_set_filesize
                                <1>
0000B9E3 8B03                   <1>       mov    eax, [ebx]
                                <1>
                                <1> loc_make_direntry_set_filesize:
0000B9E5 89471C                 <1>       mov    [edi+28], eax ; FileSize, 28
                                <1>
0000B9E8 E88AFBFFFF             <1>       call   convert_file_name
                                <1>       ;EDI = Dir Entry Format File Name Location
                                <1>       ;ESI = Dot File Name Location (capitalized)
                                <1>
0000B9ED E816000000             <1>       call   convert_current_date_time
                                <1>       ; OUTPUT -> DX = Date in dos dir entry format
                                <1>         ;        AX = Time in dos dir entry format
0000B9F2 6689470E               <1>       mov    [edi+14], ax ; CrtTime, 14
0000B9F6 66895710               <1>       mov    [edi+16], dx ; CrtDate, 16
0000B9FA 66895712               <1>       mov    [edi+18], dx ; LastAccDate, 18
0000B9FE 66894716               <1>       mov    [edi+22], ax ; WrtTime, 14
0000BA02 66895718               <1>       mov    [edi+24], dx ; WrtDate, 16
0000BA06 59                     <1>       pop    ecx
                                <1>
0000BA07 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>
0000BA08 B404                   <1>       mov    ah, 04h ; Return Current Date
0000BA0A E81DB0FFFF             <1>       call   int1Ah
                                <1>
0000BA0F 88E8                   <1>       mov    al, ch ; <- century BCD
0000BA11 240F                   <1>       and    al, 0Fh
0000BA13 88EC                   <1>       mov    ah, ch
0000BA15 C0EC04                 <1>       shr    ah, 4
0000BA18 D50A                   <1>       aad
0000BA1A 88C5                   <1>       mov    ch, al ; -> century
                                <1>
0000BA1C 88C8                   <1>       mov    al, cl ; <- year BCD
0000BA1E 240F                   <1>       and    al, 0Fh
0000BA20 88CC                   <1>       mov    ah, cl
0000BA22 C0EC04                 <1>       shr    ah, 4
0000BA25 D50A                   <1>       aad
0000BA27 88C1                   <1>       mov    cl, al ; -> year
                                <1>
0000BA29 88E8                   <1>       mov    al, ch
0000BA2B B464                   <1>       mov    ah, 100
0000BA2D F6E4                   <1>       mul    ah
0000BA2F 30ED                   <1>       xor    ch, ch
0000BA31 6601C8                 <1>       add    ax, cx
0000BA34 662DBC07               <1>       sub    ax, 1980 ; ms-dos epoch
0000BA38 6689C1                 <1>       mov    cx, ax
                                <1>
0000BA3B 88F0                   <1>       mov    al, dh ; <- month in bcd
0000BA3D 240F                   <1>       and    al, 0Fh
0000BA3F 88F4                   <1>       mov    ah, dh
0000BA41 C0EC04                 <1>       shr    ah, 4
0000BA44 D50A                   <1>       aad
0000BA46 88C6                   <1>       mov    dh, al ; -> month
```

```
1926                                    <1>
1927 0000BA48 88D0                      <1>       mov   al, dl ; <- day BCD
1928 0000BA4A 240F                      <1>       and   al, 0Fh
1929 0000BA4C 88D4                      <1>       mov   ah, dl
1930 0000BA4E C0EC04                    <1>       shr   ah, 4
1931 0000BA51 D50A                      <1>       aad
1932 0000BA53 88C2                      <1>       mov   dl, al ; -> day
1933                                    <1>
1934 0000BA55 88C8                      <1>       mov   al, cl ; count of years from 1980
1935 0000BA57 66C1E004                  <1>       shl   ax, 4
1936 0000BA5B 08F0                      <1>       or    al, dh ; month of year, 1 to 12
1937 0000BA5D 66C1E005                  <1>       shl   ax, 5
1938 0000BA61 08D0                      <1>       or    al, dl ; day of year, 1 to 31
1939                                    <1>
1940 0000BA63 6650                      <1>       push  ax ; push date
1941                                    <1>
1942 0000BA65 B402                      <1>       mov   ah, 02h ; Return Current Time
1943 0000BA67 E8C0AFFFFF                <1>       call  int1Ah
1944                                    <1>
1945 0000BA6C 88E8                      <1>       mov   al, ch ; <- hours BCD
1946 0000BA6E 240F                      <1>       and   al, 0Fh
1947 0000BA70 88EC                      <1>       mov   ah, ch
1948 0000BA72 C0EC04                    <1>       shr   ah, 4
1949 0000BA75 D50A                      <1>       aad
1950 0000BA77 88C5                      <1>       mov   ch, al ; -> hours
1951                                    <1>
1952 0000BA79 88C8                      <1>       mov   al, cl ; <- minutes BCD
1953 0000BA7B 240F                      <1>       and   al, 0Fh
1954 0000BA7D 88CC                      <1>       mov   ah, cl
1955 0000BA7F C0EC04                    <1>       shr   ah, 4
1956 0000BA82 D50A                      <1>       aad
1957 0000BA84 88C1                      <1>       mov   cl, al ; -> minutes
1958                                    <1>
1959 0000BA86 88F0                      <1>       mov   al, dh ; <- seconds BCD
1960 0000BA88 240F                      <1>       and   al, 0Fh
1961 0000BA8A 88F4                      <1>       mov   ah, dh
1962 0000BA8C C0EC04                    <1>       shr   ah, 4
1963 0000BA8F D50A                      <1>       aad
1964 0000BA91 88C6                      <1>       mov   dh, al ; -> seconds
1965                                    <1>
1966 0000BA93 88E8                      <1>       mov   al, ch ; hours
1967 0000BA95 66C1E006                  <1>       shl   ax, 6
1968 0000BA99 08C8                      <1>       or    al, cl ; minutes
1969 0000BA9B 66C1E005                  <1>       shl   ax, 5
1970 0000BA9F D0EE                      <1>       shr   dh, 1 ; 2 seconds
1971                                    <1>       ; There is a bug in TRDOS v1 here !
1972                                    <1>       ; it was 'or al, dl' !
1973 0000BAA1 08F0                      <1>       or    al, dh ; seconds
1974                                    <1>
1975 0000BAA3 665A                      <1>       pop   dx ; pop date
1976                                    <1>
1977 0000BAA5 C3                        <1>       retn
1978                                    <1>
1979                                    <1> save_directory_buffer:
1980                                    <1>       ; 15/10/2016
1981                                    <1>       ; 23/03/2016
1982                                    <1>       ; 26/02/2016
1983                                    <1>       ; 22/02/2016 (TRDOS 386 = TRDOS v2.0)
1984                                    <1>       ; 01/08/2011
1985                                    <1>       ; 14/03/2010
1986                                    <1>       ; INPUT ->
1987                                    <1>       ;      none
1988                                    <1>       ; OUTPUT ->
1989                                    <1>       ;  cf = 0 -> write OK...
1990                                    <1>       ;  cf = 1 -> error code in AL (EAX)
1991                                    <1>       ;  cf = 1 & AL = 0Dh => CH & CL = FS & FAT type
1992                                    <1>       ;  EBX = Directory Buffer Address
1993                                    <1>       ;
1994                                    <1>       ;  (EAX, ECX, EDX will be modified)
1995                                    <1>
1996 0000BAA6 BB00000800                <1>       mov   ebx, Directory_Buffer
1997 0000BAAB 803D[7C890100]02          <1>       cmp   byte [DirBuff_ValidData], 2
1998 0000BAB2 7403                      <1>       je    short loc_save_dir_buffer
1999 0000BAB4 31C0                      <1>       xor   eax, eax
2000 0000BAB6 C3                        <1>       retn
2001                                    <1>
2002                                    <1> loc_save_dir_buffer:
2003 0000BAB7 56                        <1>       push  esi
2004 0000BAB8 31DB                      <1>       xor   ebx, ebx
2005 0000BABA 8A3D[7A890100]            <1>         mov     bh, [DirBuff_DRV]
2006 0000BAC0 80EF41                    <1>       sub   bh, 'A'
2007 0000BAC3 BE00010900                <1>         mov     esi, Logical_DOSDisks
2008 0000BAC8 01DE                      <1>       add   esi, ebx
2009 0000BACA 668B4E03                  <1>         mov     cx, [esi+LD_FATType]
2010                                    <1>       ; CH = FS Type (A1h for FS)
2011                                    <1>       ; CL = FAT Type (0 for FS)
2012 0000BACE 08C9                      <1>       or    cl, cl
2013 0000BAD0 7433                      <1>       jz    short loc_save_dir_buff_stc_retn
2014                                    <1>
2015                                    <1> loc_save_dir_buffer_check_cluster_no:
2016 0000BAD2 A1[81890100]              <1>       mov   eax, [DirBuff_Cluster]
2017 0000BAD7 28FF                      <1>       sub   bh, bh ; ebx = 0
2018 0000BAD9 09C0                      <1>       or    eax, eax
2019 0000BADB 7540                      <1>       jnz   short loc_save_sub_dir_buffer
2020 0000BADD 8A25[7B890100]            <1>       mov   ah, [DirBuff_FATType]
2021 0000BAE3 FEC3                      <1>       inc   bl ;  bl = 1
2022 0000BAE5 38DC                      <1>       cmp   ah, bl
2023 0000BAE7 721D                      <1>       jb    short loc_save_dir_buff_inv_data_retn
2024 0000BAE9 FEC3                      <1>       inc   bl ; bl = 2
2025 0000BAEB 38E3                      <1>       cmp   bl, ah
2026 0000BAED 7217                      <1>       jb    short loc_save_dir_buff_inv_data_retn
2027                                    <1>
2028                                    <1> loc_save_root_dir_buffer:
2029 0000BAEF 668B5E17                  <1>       mov   bx, [esi+LD_BPB+RootDirEnts]
2030 0000BAF3 6683C30F                  <1>       add   bx, 15
```

```
2031 0000BAF7 66C1EB04            <1>       shr    bx, 4 ; 16 dir entries per sector
2032 0000BAFB 6609DB              <1>       or     bx, bx
2033 0000BAFE 7405                <1>       jz     short loc_save_dir_buff_stc_retn
2034                              <1>       ;mov   ecx, ebx
2035 0000BB00 8B4664              <1>       mov    eax, [esi+LD_ROOTBegin] ; 26/02/2016
2036 0000BB03 EB23                <1>       jmp    short loc_write_directory_to_disk
2037                              <1>
2038                              <1> loc_save_dir_buff_stc_retn:
2039 0000BB05 F9                  <1>       stc
2040                              <1> loc_save_dir_buff_inv_data_retn:
2041                              <1>       ; 15/10/2016 (0Dh -> 29)
2042 0000BB06 B01D                <1>       mov    al, 29 ; Invalid data !
2043 0000BB08 C605[7C890100]00    <1>       mov    byte [DirBuff_ValidData], 0
2044 0000BB0F EB05                <1>       jmp    short loc_save_dir_buff_retn
2045                              <1>
2046                              <1> loc_write_directory_to_disk_err:
2047                              <1>       ; 15/10/2016 (disk write error code, 1Dh -> 18)
2048 0000BB11 B812000000          <1>       mov    eax, 18 ; Drive not ready or write error
2049                              <1>
2050                              <1> loc_save_dir_buff_retn:
2051 0000BB16 BB00000800          <1>       mov    ebx, Directory_Buffer
2052 0000BB1B 5E                  <1>       pop    esi
2053 0000BB1C C3                  <1>       retn
2054                              <1>
2055                              <1> loc_save_sub_dir_buffer:
2056                              <1>       ; ebx  = 0
2057 0000BB1D 83E802              <1>       sub    eax, 2
2058 0000BB20 8A5E13              <1>       mov    bl, [esi+LD_BPB+SecPerClust]
2059 0000BB23 F7E3                <1>       mul    ebx
2060 0000BB25 034668              <1>         add     eax, [esi+LD_DATABegin]
2061                              <1>       ;mov   ecx, ebx
2062                              <1>
2063                              <1> loc_write_directory_to_disk:
2064 0000BB28 89D9                <1>       mov    ecx, ebx
2065 0000BB2A BB00000800          <1>       mov    ebx, Directory_Buffer
2066 0000BB2F E8DA670000          <1>       call   disk_write
2067 0000BB34 72DB                <1>       jc     short loc_write_directory_to_disk_err
2068                              <1>
2069                              <1> loc_save_dir_buff_validate_retn:
2070 0000BB36 C605[7C890100]01    <1>       mov    byte [DirBuff_ValidData], 1
2071 0000BB3D 31C0                <1>       xor    eax, eax
2072                              <1>       ; 26/02/2016
2073 0000BB3F EBD5                <1>       jmp    short loc_save_dir_buff_retn
2074                              <1>
2075                              <1> update_parent_dir_lmdt:
2076                              <1>       ; 29/12/2017
2077                              <1>       ; 22/02/2016 (TRDOS 386 = TRDOS v2.0)
2078                              <1>       ; 01/08/2011
2079                              <1>       ; 16/10/2010
2080                              <1>       ;
2081                              <1>       ; INPUT ->
2082                              <1>       ;      none
2083                              <1>       ; OUTPUT ->
2084                              <1>       ;      (last modification date & time of the parent dir
2085                              <1>       ;      will be changed/updated)
2086                              <1>       ;
2087                              <1>       ; (EAX, EBX, ECX, EDX, EDI will be changed)
2088                              <1>
2089 0000BB41 29C0                <1>       sub    eax, eax
2090 0000BB43 8A25[54820100]      <1>       mov    ah, [Current_Dir_Level]
2091 0000BB49 A0[55820100]        <1>       mov    al, [Current_FATType]
2092 0000BB4E 3C01                <1>       cmp    al, 1
2093 0000BB50 723A                <1>       jb     short loc_UPDLMDT_proc_retn
2094                              <1>
2095                              <1> loc_update_parent_dir_lm_date_time:
2096 0000BB52 08E4                <1>       or     ah, ah
2097 0000BB54 7436                <1>       jz     short loc_UPDLMDT_proc_retn
2098                              <1>
2099 0000BB56 56                  <1>       push   esi ; *
2100 0000BB57 8825[F48B0100]      <1>       mov    [UPDLMDT_CDirLevel], ah
2101 0000BB5D 8B15[50820100]      <1>       mov    edx, [Current_Dir_FCluster]
2102 0000BB63 8915[F58B0100]      <1>       mov    [UPDLMDT_CDirFCluster], edx
2103                              <1>
2104 0000BB69 FECC                <1>       dec    ah
2105 0000BB6B B90C000000          <1>       mov    ecx, 12
2106 0000BB70 BE[B3890100]        <1>         mov     esi, PATH_Array
2107                              <1>
2108 0000BB75 8825[54820100]      <1>       mov    [Current_Dir_Level], ah
2109 0000BB7B 08E4                <1>       or     ah, ah
2110 0000BB7D 750E                <1>       jnz    short loc_update_parent_dir_lmdt_load_sub_dir_1
2111 0000BB7F 803D[55820100]02    <1>       cmp    byte [Current_FATType], 2
2112 0000BB86 770B                <1>       ja     short loc_update_parent_dir_lmdt_load_sub_dir_2
2113 0000BB88 28C0                <1>       sub    al, al ; eax = 0
2114 0000BB8A EB0A                <1>       jmp    short loc_update_parent_dir_lmdt_load_sub_dir_3
2115                              <1>
2116                              <1> loc_UPDLMDT_proc_retn:
2117 0000BB8C C3                  <1>       retn
2118                              <1>
2119                              <1> loc_update_parent_dir_lmdt_load_sub_dir_1:
2120 0000BB8D B010                <1>       mov    al, 16
2121 0000BB8F F6E4                <1>       mul    ah
2122 0000BB91 01C6                <1>       add    esi, eax
2123                              <1>
2124                              <1> loc_update_parent_dir_lmdt_load_sub_dir_2:
2125 0000BB93 8B460C              <1>       mov    eax, [esi+12] ; Parent Dir First Cluster
2126                              <1>
2127                              <1> loc_update_parent_dir_lmdt_load_sub_dir_3:
2128 0000BB96 A3[50820100]        <1>       mov    [Current_Dir_FCluster], eax
2129                              <1>
2130 0000BB9B 83C610              <1>       add    esi, 16
2131 0000BB9E 66BF[DA8A]          <1>       mov    di, Dir_File_Name
2132 0000BBA2 F3A4                <1>       rep    movsb
2133                              <1>
2134 0000BBA4 BE00010900          <1>       mov    esi, Logical_DOSDisks
2135 0000BBA9 29DB                <1>       sub    ebx, ebx
```

```
2136 0000BBAB 8A3D[56820100]      <1>       mov    bh, [Current_Drv]
2137 0000BBB1 01DE                <1>       add    esi, ebx
2138 0000BBB3 E88FF7FFFF          <1>       call   reload_current_directory
2139 0000BBB8 7230                <1>       jc     short loc_update_parent_dir_lmdt_restore_cdirlevel
2140                              <1>
2141                              <1> loc_update_parent_dir_lmdt_locate_dir:
2142 0000BBBA BE[DA8A0100]        <1>       mov    esi, Dir_File_Name
2143 0000BBBF 6631C9              <1>       xor    cx, cx
2144 0000BBC2 66B81008            <1>       mov    ax, 0810h ; Only directories
2145 0000BBC6 E8B5F6FFFF          <1>         call    locate_current_dir_file
2146                              <1>       ; EDI = DirBuff Directory Entry Address
2147 0000BBCB 721D                <1>       jc     short loc_update_parent_dir_lmdt_restore_cdirlevel
2148                              <1>
2149 0000BBCD E836FEFFFF          <1>       call   convert_current_date_time
2150 0000BBD2 66895712            <1>       mov    [edi+18], dx ; Last Access Date
2151 0000BBD6 66895718            <1>       mov    [edi+24], dx ; Last Write Date
2152 0000BBDA 66894716            <1>       mov    [edi+22], ax ; Last Write Time
2153                              <1>
2154 0000BBDE C605[7C890100]02    <1>       mov    byte [DirBuff_ValidData], 2
2155 0000BBE5 E8BCFEFFFF          <1>       call   save_directory_buffer
2156                              <1>       ; 29/12/2017
2157                              <1>       ;jc    short loc_update_parent_dir_lmdt_restore_cdirlevel
2158                              <1>       ;xor   al, al
2159                              <1> loc_update_parent_dir_lmdt_restore_cdirlevel:
2160                              <1>       ;current directory level restoration
2161 0000BBEA 8A25[F48B0100]      <1>       mov    ah, [UPDLMDT_CDirLevel]
2162 0000BBF0 8825[54820100]      <1>       mov    [Current_Dir_Level], ah
2163 0000BBF6 8B15[F58B0100]      <1>         mov     edx, [UPDLMDT_CDirFCluster]
2164 0000BBFC 8915[50820100]      <1>       mov    [Current_Dir_FCluster], edx
2165                              <1>
2166 0000BC02 5E                  <1>       pop    esi ; *
2167 0000BC03 C3                  <1>       retn
2168                              <1>
2169                              <1> delete_longname:
2170                              <1>       ; 27/02/2016 (TRDOS 386 = TRDOS v2.0)
2171                              <1>       ; 01/08/2011 (DIR.ASM, 'proc_delete_longname')
2172                              <1>       ; 14/03/2010
2173                              <1>       ; INPUT ->
2174                              <1>       ;      EAX = Directory Entry (Index) Number (< 65536)
2175                              <1>       ; OUTPUT ->
2176                              <1>       ;      cf = 0 -> OK  (EAX = 0)
2177                              <1>       ;      cf = 1 -> error code in EAX (AL)
2178                              <1>       ;
2179                              <1>       ; (Modified registers: EAX, EDX, ECX, EBX, EDI)
2180                              <1>
2181 0000BC04 66A3[248C0100]      <1>       mov    [DLN_EntryNumber], ax
2182 0000BC0A C605[268C0100]40    <1>         mov     byte [DLN_40h], 40h
2183                              <1>
2184 0000BC11 E858000000          <1>       call   locate_current_dir_entry
2185 0000BC16 7308                <1>       jnc    short loc_dln_check_attributes
2186 0000BC18 C3                  <1>       retn
2187                              <1>
2188                              <1> loc_dln_longname_not_found:
2189 0000BC19 B802000000          <1>       mov    eax, 2
2190 0000BC1E F9                  <1>       stc
2191 0000BC1F C3                  <1>       retn
2192                              <1>
2193                              <1> loc_dln_check_attributes:
2194 0000BC20 B00F                <1>       mov    al, 0Fh  ; long name
2195 0000BC22 8A670B              <1>       mov    ah, [edi+0Bh] ; dir entry attributes
2196 0000BC25 38C4                <1>       cmp    ah, al
2197 0000BC27 75F0                <1>       jne    short loc_dln_longname_not_found
2198 0000BC29 8A27                <1>       mov    ah, [edi]
2199 0000BC2B 2A25[268C0100]      <1>       sub    ah, [DLN_40h]
2200 0000BC31 76E6                <1>       jna    short loc_dln_longname_not_found
2201 0000BC33 80FC14              <1>       cmp    ah, 14h ; 84-64=20 -> 20*13=260 bytes
2202 0000BC36 77E1                <1>       ja     short loc_dln_longname_not_found
2203                              <1>
2204 0000BC38 C607E5              <1>       mov    byte [edi], 0E5h  ; deleted sign
2205 0000BC3B C605[7C890100]02    <1>       mov    byte [DirBuff_ValidData], 2 ; changed/write sign
2206 0000BC42 C605[268C0100]00    <1>       mov    byte [DLN_40h], 0 ; 40h -> 0
2207                              <1>
2208                              <1> loc_dln_delete_next_ln_entry:
2209 0000BC49 80FC01              <1>       cmp    ah, 1
2210 0000BC4C 7616                <1>       jna    short loc_dln_longname_retn
2211                              <1> loc_dln_delete_next_ln_entry_0:
2212 0000BC4E 66FF05[248C0100]    <1>       inc    word [DLN_EntryNumber]
2213 0000BC55 0FB705[248C0100]    <1>       movzx  eax, word [DLN_EntryNumber]
2214 0000BC5C E80D000000          <1>       call   locate_current_dir_entry
2215 0000BC61 73BD                <1>       jnc    short loc_dln_check_attributes
2216                              <1>
2217                              <1> loc_dln_longname_stc_retn:
2218 0000BC63 C3                  <1>       retn
2219                              <1>
2220                              <1> loc_dln_longname_retn:
2221                              <1>       ;cmp   byte [DirBuff_ValidData], 2
2222                              <1>       ;jne   short loc_dln_longname_retn_xor_eax
2223 0000BC64 E83DFEFFFF          <1>       call   save_directory_buffer
2224 0000BC69 72F8                <1>       jc     short loc_dln_longname_stc_retn
2225                              <1>
2226                              <1> loc_dln_longname_retn_xor_eax:
2227 0000BC6B 31C0                <1>       xor    eax, eax
2228 0000BC6D C3                  <1>       retn
2229                              <1>
2230                              <1> locate_current_dir_entry:
2231                              <1>       ; 16/10/2016
2232                              <1>       ; 15/10/2016
2233                              <1>       ; 23/03/2016
2234                              <1>       ; 27/02/2016 (TRDOS 386 = TRDOS v2.0)
2235                              <1>       ; 01/08/2011 (DIR.ASM, 'proc_locate_current_dir_entry')
2236                              <1>       ; 07/03/2010
2237                              <1>       ; INPUT ->
2238                              <1>       ;      EAX = Directory Entry (Index) Number (< 65536)
2239                              <1>       ; OUTPUT ->
2240                              <1>       ;      EDI = Directory Entry Address
```

```
2241                            <1>       ;      EAX = Cluster Number of Directory Buffer
2242                            <1>       ;      EBX = Directory Buffer Entry Offset
2243                            <1>       ;      ECX = DirBuff Valid Data identifier (CL)
2244                            <1>       ;      If CF = 0 and CL = 2 then
2245                            <1>       ;         directory buffer modified and
2246                            <1>       ;         must be written to disk.
2247                            <1>       ;      If CF = 0  and CL = 1 then
2248                            <1>       ;         dir buffer has been written to disk, already.
2249                            <1>       ;      CF = 1 -> Error code in EAX (AL)
2250                            <1>       ;
2251                            <1>       ; (Modified registers: EAX, EDX, ECX, EBX, EDI)
2252                            <1>
2253                            <1> loc_locate_current_dir_entry:
2254 0000BC6E 56               <1>       push   esi
2255 0000BC6F 89C1             <1>       mov    ecx, eax
2256 0000BC71 BA20000000       <1>       mov    edx, 32
2257 0000BC76 F7E2             <1>       mul    edx
2258 0000BC78 A3[308C0100]     <1>       mov    [LCDE_ByteOffset], eax
2259 0000BC7D 31DB             <1>       xor    ebx, ebx
2260 0000BC7F 8A3D[56820100]   <1>       mov    bh, [Current_Drv]
2261 0000BC85 A0[7A890100]     <1>         mov     al, [DirBuff_DRV]
2262 0000BC8A 2C41             <1>       sub    al, 'A'
2263 0000BC8C BE00010900       <1>         mov     esi, Logical_DOSDisks
2264 0000BC91 01DE             <1>       add    esi, ebx
2265 0000BC93 38C7             <1>       cmp    bh, al
2266 0000BC95 0F8592000000     <1>         jne     loc_lcde_reload_current_directory
2267                            <1> loc_lcde_cdl_check:
2268 0000BC9B 803D[54820100]00 <1>       cmp    byte [Current_Dir_Level], 0
2269 0000BCA2 772A             <1>       ja     short loc_lcde_calc_dirbuff_cluster_offset
2270                            <1>       ; 27/02/2016
2271                            <1>       ; TRDOS v1 has bug here for FAT32 fs !
2272                            <1>       ; (Root Directory Entries for FAT32 = 0)
2273 0000BCA4 807E0303           <1>       cmp    byte [esi+LD_FATType], 3  ; FAT32
2274 0000BCA8 7324             <1>       jnb    short loc_lcde_calc_dirbuff_cluster_offset
2275                            <1>
2276                            <1> loc_lcde_cdl_check_FAT12_16:
2277 0000BCAA 668B4617         <1>       mov    ax, [esi+LD_BPB+RootDirEnts]
2278 0000BCAE 6648             <1>       dec    ax
2279                            <1>       ;xor   dx, dx
2280 0000BCB0 6639C8           <1>       cmp    ax, cx ; cx = Directory Entry (Index) Number
2281 0000BCB3 720E             <1>       jb     short loc_lcde_stc_12h_retn
2282 0000BCB5 66890D[288C0100] <1>       mov    [LCDE_EntryIndex], cx
2283 0000BCBC 31C0             <1>       xor    eax, eax
2284 0000BCBE E993000000       <1>         jmp     loc_lcde_check_dir_buffer_cluster
2285                            <1>
2286                            <1> loc_lcde_stc_12h_retn:
2287 0000BCC3 5E               <1>       pop    esi
2288 0000BCC4 89CB             <1>       mov    ebx, ecx
2289 0000BCC6 89D1             <1>       mov    ecx, edx
2290                            <1>       ; 16/10/2016 (12h -> 12)
2291 0000BCC8 B80C000000       <1>       mov    eax, 12 ; No more files
2292 0000BCCD C3               <1>       retn
2293                            <1>
2294                            <1> loc_lcde_calc_dirbuff_cluster_offset:
2295 0000BCCE 8A5E13           <1>       mov    bl, [esi+LD_BPB+SecPerClust]
2296 0000BCD1 30FF             <1>       xor    bh, bh
2297 0000BCD3 668B4611         <1>       mov    ax, [esi+LD_BPB+BytesPerSec]
2298 0000BCD7 66F7E3           <1>       mul    bx
2299 0000BCDA 6609D2           <1>       or     dx, dx ; If bytes per cluster > 32KB it is invalid
2300 0000BCDD 755D             <1>       jnz    short loc_lcde_invalid_format
2301                            <1>       ;mov   ecx, eax
2302 0000BCDF 6689C1           <1>       mov    cx, ax ; BYTES PER CLUSTER
2303 0000BCE2 A1[308C0100]     <1>       mov    eax, [LCDE_ByteOffset]
2304                            <1>       ;sub   edx, edx
2305 0000BCE7 F7F1             <1>       div    ecx
2306 0000BCE9 3DFFFF0000       <1>       cmp    eax, 65535
2307 0000BCEE 774C             <1>       ja     short loc_lcde_invalid_format
2308                            <1>
2309                            <1>       ; cluster sequence number of directory (< 65536)
2310 0000BCF0 66A3[2A8C0100]   <1>       mov    [LCDE_ClusterSN], ax
2311                            <1>
2312 0000BCF6 6689D0           <1>       mov    ax, dx ; byte offset in cluster (directory buffer)
2313 0000BCF9 66BB2000         <1>       mov    bx, 32 ; ; 1 dir entry = 32 bytes
2314 0000BCFD 6629D2           <1>         sub     dx, dx  ; 0
2315 0000BD00 66F7F3           <1>       div    bx
2316 0000BD03 66A3[288C0100]   <1>       mov    [LCDE_EntryIndex], ax ; dir entry index/sequence number
2317                            <1>                                    ; (in directory buffer/cluster)
2318                            <1> loc_lcde_get_current_sub_dir_fcluster:
2319 0000BD09 A1[50820100]     <1>       mov    eax, [Current_Dir_FCluster]
2320                            <1>
2321                            <1> loc_lcde_get_next_cluster:
2322 0000BD0E 66833D[2A8C0100]00 <1>       cmp    word [LCDE_ClusterSN], 0
2323 0000BD16 763E             <1>       jna    short loc_lcde_check_dir_buffer_cluster
2324 0000BD18 A3[2C8C0100]     <1>       mov    [LCDE_Cluster], eax
2325 0000BD1D E834100000       <1>       call   get_next_cluster
2326 0000BD22 7220             <1>       jc     short loc_lcde_check_gnc_error
2327 0000BD24 66FF0D[2A8C0100] <1>       dec    word [LCDE_ClusterSN]
2328 0000BD2B EBE1             <1>       jmp    short loc_lcde_get_next_cluster
2329                            <1>
2330                            <1> loc_lcde_reload_current_directory:
2331 0000BD2D 51               <1>       push   ecx
2332 0000BD2E E814F6FFFF       <1>       call   reload_current_directory
2333 0000BD33 59               <1>       pop    ecx
2334 0000BD34 0F8361FFFFFF     <1>         jnc     loc_lcde_cdl_check
2335 0000BD3A 5E               <1>       pop    esi
2336 0000BD3B C3               <1>       retn
2337                            <1>
2338                            <1> loc_lcde_invalid_format:
2339                            <1>       ; 15/10/2016 (0Bh -> 28)
2340 0000BD3C B81C000000       <1>       mov    eax, 28 ; Invalid Format !
2341                            <1> loc_lcde_drive_not_ready_read_err:
2342 0000BD41 F9               <1>       stc
2343 0000BD42 5E               <1>       pop    esi
2344 0000BD43 C3               <1>       retn
2345                            <1>
```

```
                                <1> loc_lcde_check_gnc_error:
0000BD44 09C0                   <1>         or      eax, eax
0000BD46 75F9                   <1>         jnz     short loc_lcde_drive_not_ready_read_err
0000BD48 66FF0D[2A8C0100]       <1>         dec     word [LCDE_ClusterSN]
0000BD4F 75EB                   <1>         jnz     short loc_lcde_invalid_format
0000BD51 A1[2C8C0100]           <1>         mov     eax, [LCDE_Cluster]
                                <1>
                                <1> loc_lcde_check_dir_buffer_cluster:
0000BD56 3B05[81890100]         <1>         cmp     eax, [DirBuff_Cluster]
0000BD5C 755C                   <1>         jne     short loc_lcde_load_dir_cluster
0000BD5E 803D[7C890100]00       <1>         cmp     byte [DirBuff_ValidData], 0
0000BD65 7727                   <1>         ja      short lcde_check_dir_buffer_cluster_next
0000BD67 803D[54820100]00       <1>         cmp     byte [Current_Dir_Level], 0
0000BD6E 775F                   <1>         ja      short loc_lcde_load_dir_cluster_0
                                <1>         ; 27/02/2016
                                <1>         ; TRDOS v1 has bug here for FAT32 fs !
0000BD70 807E0303               <1>         cmp     byte [esi+LD_FATType], 3  ; FAT32
0000BD74 7359                   <1>         jnb     short loc_lcde_load_dir_cluster_0
                                <1>         ;
0000BD76 0FB74E17               <1>         movzx   ecx, word [esi+LD_BPB+RootDirEnts]
0000BD7A 6683C10F               <1>         add     cx, 15 ; round up (16 entries per sector)
0000BD7E 66C1E904               <1>         shr     cx, 4 ; 1 sector contains 16 dir entries
                                <1>
0000BD82 8B4664                 <1>           mov     eax, [esi+LD_ROOTBegin]
0000BD85 EB54                   <1>         jmp     short loc_lcde_load_dir_cluster_1
                                <1>
                                <1> loc_lcde_validate_dirBuff:
0000BD87 C605[7C890100]01       <1>         mov     byte [DirBuff_ValidData], 1
                                <1>
                                <1> lcde_check_dir_buffer_cluster_next:
0000BD8E 0FB71D[288C0100]       <1>         movzx   ebx, word [LCDE_EntryIndex]
0000BD95 663B1D[7F890100]       <1>         cmp     bx, [DirBuff_LastEntry]
0000BD9C 779E                   <1>         ja      short loc_lcde_invalid_format
0000BD9E B820000000             <1>         mov     eax, 32
0000BDA3 F7E3                   <1>         mul     ebx
                                <1>         ;or     edx, edx
                                <1>         ;jnz    short loc_lcde_invalid_format
                                <1>
0000BDA5 BF00000800             <1>         mov     edi, Directory_Buffer
0000BDAA 01C7                   <1>         add     edi, eax ; add entry offset to buffer address
                                <1>
                                <1> loc_lcde_dir_buffer_last_check:
0000BDAC A1[81890100]           <1>         mov     eax, [DirBuff_Cluster]
0000BDB1 0FB60D[7C890100]       <1>         movzx   ecx, byte [DirBuff_ValidData]
                                <1>
                                <1> loc_lcde_retn:
0000BDB8 5E                     <1>         pop     esi
0000BDB9 C3                     <1>         retn
                                <1>
                                <1> loc_lcde_load_dir_cluster:
                                <1>         ;cmp    byte [DirBuff_ValidData], 2
                                <1>         ;jne    short loc_lcde_load_dir_cluster_n2
0000BDBA 50                     <1>         push    eax
0000BDBB E8E6FCFFFF             <1>         call    save_directory_buffer
0000BDC0 58                     <1>         pop     eax
0000BDC1 72F5                   <1>         jc      short loc_lcde_retn
                                <1>
                                <1> loc_lcde_load_dir_cluster_n2:
0000BDC3 C605[7C890100]00       <1>         mov     byte [DirBuff_ValidData], 0
0000BDCA A3[81890100]           <1>         mov     [DirBuff_Cluster], eax
                                <1>
                                <1> loc_lcde_load_dir_cluster_0:
0000BDCF 83E802                 <1>         sub     eax, 2
0000BDD2 0FB64E13               <1>         movzx   ecx, byte [esi+LD_BPB+SecPerClust]
0000BDD6 F7E1                   <1>         mul     ecx
0000BDD8 034668                 <1>           add     eax, [esi+LD_DATABegin]
                                <1>
                                <1> loc_lcde_load_dir_cluster_1:
0000BDDB BB00000800             <1>         mov     ebx, Directory_Buffer
                                <1>         ; ecx = sector count
0000BDE0 E838650000             <1>         call    disk_read
0000BDE5 73A0                   <1>         jnc     short loc_lcde_validate_dirBuff
                                <1>
                                <1>         ; 15/10/2016
                                <1>         ; (Disk read error instead of drv not ready err)
0000BDE7 B811000000             <1>         mov     eax, 17 ; Drive not ready or read error !
0000BDEC 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>
0000BDEE 29C0                   <1>         sub     eax, eax
0000BDF0 88FC                   <1>         mov     ah, bh
0000BDF2 BE00010900             <1>         mov     esi, Logical_DOSDisks
0000BDF7 01C6                   <1>         add     esi, eax
                                <1>
0000BDF9 807E0301               <1>         cmp     byte [esi+LD_FATType], 1
0000BDFD 7312                   <1>         jnb     short loc_del_fat_file
                                <1>
0000BDFF 807E04A1               <1>         cmp     byte [esi+LD_FSType], 0A1h
0000BE03 7406                   <1>         je      short loc_del_fs_file
                                <1>
                                <1> loc_del_file_invalid_format:
0000BE05 30E4                   <1>         xor     ah, ah
                                <1>         ; 15/10/2016 (0Bh -> 28)
0000BE07 B01C                   <1>         mov     al, 28  ; Invalid Format
```

```
0000BE09 F9                <1>       stc
0000BE0A C3                <1>       retn
                           <1>
                           <1> loc_del_fs_file:
0000BE0B E83F0F0000        <1>       call   delete_fs_file
0000BE10 C3                <1>       retn
                           <1>
                           <1> loc_del_fat_file:
0000BE11 E808000000        <1>       call   delete_directory_entry
0000BE16 7205              <1>       jc     short loc_del_file_err_retn
                           <1>
                           <1> loc_delfile_unlink_cluster_chain:
0000BE18 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:
0000BE1D 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>
0000BE1E 881D[BE8B0100]    <1>       mov    [DelFile_LNEL], bl
0000BE24 66890D[BC8B0100]  <1>       mov    [DelFile_EntryCounter], cx
                           <1>
0000BE2B 668B4714          <1>       mov    ax, [edi+20] ; First Cluster High Word
0000BE2F C1E010            <1>       shl    eax, 16
0000BE32 668B471A          <1>       mov    ax, [edi+26] ; First Cluster Low Word
                           <1>
0000BE36 A3[B88B0100]      <1>       mov    [DelFile_FCluster], eax
                           <1>
                           <1> loc_del_short_name:
0000BE3B C607E5            <1>       mov    byte [edi], 0E5h  ; Deleted sign
                           <1>
0000BE3E C605[7C890100]02  <1>       mov    byte [DirBuff_ValidData], 2
0000BE45 E85CFCFFFF        <1>       call   save_directory_buffer
0000BE4A 723D              <1>       jc     short loc_delete_direntry_err_return
                           <1>
                           <1> loc_del_long_name:
0000BE4C 0FB615[BE8B0100]  <1>       movzx  edx, byte [DelFile_LNEL]
0000BE53 08D2              <1>       or     dl, dl
0000BE55 7416              <1>       jz     short loc_del_dir_entry_update_parent_dir_lm_date
                           <1>
0000BE57 8835[BE8B0100]    <1>       mov    byte [DelFile_LNEL], dh ; 0
                           <1>
0000BE5D 0FB705[BC8B0100]  <1>       movzx  eax,  word [DelFile_EntryCounter]
0000BE64 29D0              <1>       sub    eax, edx
                           <1>       ;jnc   short loc_del_long_name_continue
0000BE66 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
0000BE68 E897FDFFFF        <1>       call   delete_longname
                           <1>       ;jc    short loc_delete_direntry_err_return
                           <1>
                           <1> loc_del_dir_entry_update_parent_dir_lm_date:
0000BE6D 801D[BE8B0100]00  <1>       sbb    byte [DelFile_LNEL], 0 ; 0FFh if cf = 1
                           <1>
0000BE74 E8C8FCFFFF        <1>       call   update_parent_dir_lmdt
0000BE79 B700              <1>       mov    bh, 0
0000BE7B 80D700            <1>       adc    bh, 0
                           <1>
0000BE7E 8A1D[BE8B0100]    <1>       mov    bl, byte [DelFile_LNEL]
                           <1>
                           <1> loc_delete_direntry_return:
0000BE84 A1[B88B0100]      <1>       mov    eax, [DelFile_FCluster]
                           <1> loc_delete_direntry_err_return:
0000BE89 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>       ;
```

```
2556                            <1>       ; (EAX, EBX, ECX, EDX, ESI, EDI will be changed)
2557                            <1>
2558 0000BE8A 803D[55820100]00 <1>       cmp    byte [Current_FATType], 0
2559 0000BE91 7706             <1>       ja     short loc_rename_directory_entry
2560                            <1>
2561 0000BE93 E8B80E0000       <1>       call   rename_fs_file_or_directory
2562 0000BE98 C3               <1>       retn
2563                            <1>
2564                            <1> loc_rename_directory_entry:
2565 0000BE99 881D[BE8B0100]   <1>       mov    [DelFile_LNEL], bl
2566 0000BE9F 66890D[BC8B0100] <1>       mov    [DelFile_EntryCounter], cx
2567 0000BEA6 A3[B88B0100]     <1>       mov    [DelFile_FCluster], eax
2568                            <1>
2569 0000BEAB 0FB7C1           <1>       movzx  eax, cx
2570 0000BEAE E8BBFDFFFF       <1>       call   locate_current_dir_entry
2571 0000BEB3 7308             <1>       jnc    short loc_rename_direntry_check_fcluster
2572                            <1>
2573                            <1> loc_rename_direntry_pop_retn:
2574 0000BEB5 C3               <1>       retn
2575                            <1>
2576                            <1> loc_rename_direntry_pop_invd_retn:
2577 0000BEB6 F9               <1>       stc
2578                            <1> loc_rename_direntry_invd_retn:
2579                            <1>       ; 15/10/2016 (0Dh -> 29)
2580 0000BEB7 B81D000000       <1>       mov    eax, 29 ; Invalid data
2581                            <1> loc_rename_retn:
2582 0000BEBC C3               <1>       retn
2583                            <1>
2584                            <1> loc_rename_direntry_check_fcluster:
2585 0000BEBD 668B5714         <1>       mov    dx, [edi+20] ; First Cluster HW
2586 0000BEC1 C1E210           <1>       shl    edx, 16 ; 13/11/2017
2587 0000BEC4 668B571A         <1>       mov    dx, [edi+26] ; First Cluster LW
2588 0000BEC8 3B15[B88B0100]   <1>       cmp    edx, [DelFile_FCluster]
2589 0000BECE 75E6             <1>       jne    short loc_rename_direntry_pop_invd_retn
2590                            <1>       ; ESI = New file (or directory) name. (ASCIIZ string)
2591                            <1>       ; 06/03/2016
2592                            <1>       ; TRDOS v2 - NOTE: 'convert_file_name' procedure
2593                            <1>       ; has been modified for eliminating following situation.
2594                            <1>       ;
2595                            <1>       ; TRDOS v1 - NOTE: If file/dir name is more than 11 bytes
2596                            <1>       ; without a dot, attributes (edi+11) byte will be overwritten !
2597                            <1>       ; (Dot file name input must be proper for 11 byte dir entry
2598                            <1>       ;  type file name output.)
2599 0000BED0 E8A2F6FFFF       <1>       call   convert_file_name
2600                            <1>
2601 0000BED5 C605[7C890100]02 <1>         mov     byte [DirBuff_ValidData], 2
2602 0000BEDC E8C5FBFFFF       <1>       call   save_directory_buffer
2603 0000BEE1 72D9             <1>       jc     short loc_rename_retn
2604                            <1>
2605                            <1> loc_rename_direntry_del_ln:
2606 0000BEE3 0FB615[BE8B0100] <1>       movzx  edx, byte [DelFile_LNEL]
2607 0000BEEA 08D2             <1>       or     dl, dl
2608 0000BEEC 7410             <1>       jz     short loc_rename_direntry_update_parent_dir_lm_date
2609                            <1>
2610 0000BEEE 0FB705[BC8B0100] <1>       movzx  eax, word [DelFile_EntryCounter]
2611 0000BEF5 29D0             <1>       sub    eax, edx
2612 0000BEF7 72BE             <1>       jc     short loc_rename_direntry_invd_retn
2613                            <1>
2614                            <1> loc_rename_direntry_del_ln_continue:
2615                            <1>       ; EAX = Directory Entry Number of the long name last entry
2616 0000BEF9 E806FDFFFF       <1>       call   delete_longname
2617                            <1>
2618                            <1> loc_rename_direntry_update_parent_dir_lm_date:
2619 0000BEFE E83EFCFFFF       <1>       call   update_parent_dir_lmdt
2620 0000BF03 31C0             <1>       xor    eax, eax
2621 0000BF05 C3               <1>       retn
2622                            <1>
2623                            <1> move_source_file_to_destination_file:
2624                            <1>       ; 15/10/2016
2625                            <1>       ; 11/03/2016
2626                            <1>       ; 10/03/2016 (TRDOS 386 = TRDOS v2.0)
2627                            <1>       ; 01/08/2011 (FILE.ASM)
2628                            <1>       ; 04/08/2010
2629                            <1>       ;
2630                            <1>       ;   Phase 1 -> Check destination file,
2631                            <1>       ;              'not found' is required
2632                            <1>       ;   Phase 2 -> Check source file
2633                            <1>       ;              'found' and proper attributes is required
2634                            <1>       ;   Phase 3 -> Make destination directory entry,
2635                            <1>       ;            add new dir cluster or section if it is required
2636                            <1>       ;   Phase 4 -> Delete source directory entry.
2637                            <1>       ;       cf = 1 causes to return before the phase 4.
2638                            <1>       ;    (source file protection against any possible errors)
2639                            <1>       ;
2640                            <1>       ; 08/05/2011 major modification
2641                            <1>       ;            -> destination file deleting is removed
2642                            <1>       ;            for msdos move/rename compatibility.
2643                            <1>       ;            (Access denied error will return if
2644                            <1>       ;            the destination file is found...)
2645                            <1>       ; INPUT ->
2646                            <1>       ;      ESI = Source File Pathname (Asciiz)
2647                            <1>       ;        EDI = Destination File Pathname (Asciiz)
2648                            <1>       ;        AL = 0 --> Interrupt (System call)
2649                            <1>       ;        AL > 0 --> Command Interpreter (Question)
2650                            <1>       ;        AL = 1 --> Question Phase
2651                            <1>       ;        AL = 2 --> Progress Phase
2652                            <1>       ; OUTPUT ->
2653                            <1>       ;      cf = 0 -> OK
2654                            <1>       ;        EAX = Destination directory first cluster
2655                            <1>       ;        ESI = Logical DOS drive description table
2656                            <1>       ;        EBX = Destination file structure offset
2657                            <1>       ;        CX = 0 (CX > 0 --> calculate free space error)
2658                            <1>       ;        cf = 1 -> Error code in EAX (AL)
2659                            <1>       ;
2660                            <1>       ;  (EDX, ECX, EBX, ESI, EDI will be changed)
```

```
2661                                    <1>
2662 0000BF06 3C02                     <1>       cmp    al, 2
2663 0000BF08 0F847F010000             <1>       je     msftdf_df2_check_directory
2664 0000BF0E A2[3E8D0100]             <1>       mov    [move_cmd_phase], al
2665                                    <1>
2666                                    <1> msftdf_parse_sf_path:
2667                                    <1>       ; ESI = ASCIIZ pathname (Source)
2668 0000BF13 57                       <1>       push   edi
2669 0000BF14 BF[3C8C0100]             <1>       mov    edi, SourceFile_Drv
2670 0000BF19 E824F7FFFF               <1>       call   parse_path_name
2671 0000BF1E 5E                       <1>       pop    esi
2672 0000BF1F 7211                     <1>       jc     short msftdf_psf_retn
2673                                    <1>
2674                                    <1> msftdf_parse_df_path:
2675                                    <1>       ; ESI = ASCIIZ pathname   (Destination)
2676 0000BF21 BF[BC8C0100]             <1>       mov    edi, DestinationFile_Drv
2677 0000BF26 E817F7FFFF               <1>       call   parse_path_name
2678 0000BF2B 7306                     <1>       jnc    short msftdf_check_sf_drv
2679                                    <1>
2680 0000BF2D 3C01                     <1>       cmp    al, 1 ; File or directory name is not existing
2681 0000BF2F 7602                     <1>       jna    short msftdf_check_sf_drv
2682                                    <1>
2683                                    <1> msftdf_stc_retn:
2684 0000BF31 F9                       <1>       stc
2685                                    <1> msftdf_psf_retn:
2686 0000BF32 C3                       <1>       retn
2687                                    <1>
2688                                    <1> msftdf_check_sf_drv:
2689 0000BF33 A0[3C8C0100]             <1>       mov    al, [SourceFile_Drv]
2690                                    <1>
2691                                    <1> msftdf_check_df_drv:
2692 0000BF38 8A15[BC8C0100]           <1>       mov    dl, [DestinationFile_Drv]
2693                                    <1>
2694                                    <1> msftdf_compare_sf_df_drv:
2695 0000BF3E 29DB                     <1>       sub    ebx, ebx
2696 0000BF40 8A3D[56820100]           <1>       mov    bh, [Current_Drv]
2697 0000BF46 38C2                     <1>       cmp    dl, al
2698 0000BF48 7409                     <1>       je     short msftdf_check_sf_df_drv_ok
2699                                    <1>
2700                                    <1> msftdf_not_same_drv:
2701                                    <1>         ; DL = source file's drive number
2702 0000BF4A 88C6                     <1>       mov    dh, al ; destination file's drive number
2703                                    <1>       ; 15/10/2016 (11h -> 21)
2704 0000BF4C B815000000               <1>       mov    eax, 21 ; Not the same drive
2705 0000BF51 F9                       <1>       stc
2706 0000BF52 C3                       <1>       retn
2707                                    <1>
2708                                    <1> msftdf_check_sf_df_drv_ok:
2709 0000BF53 8815[3F8D0100]           <1>       mov    [msftdf_sf_df_drv], dl
2710                                    <1>
2711 0000BF59 29C0                     <1>         sub  eax, eax
2712 0000BF5B 88D4                     <1>       mov    ah, dl
2713 0000BF5D 0500010900               <1>       add    eax, Logical_DOSDisks
2714 0000BF62 A3[408D0100]             <1>       mov    [msftdf_drv_offset], eax
2715                                    <1>
2716 0000BF67 38FA                     <1>       cmp    dl, bh ; byte [Current_Drv]
2717 0000BF69 7407                     <1>       je     short msftdf_df_check_directory
2718                                    <1>
2719                                    <1> msftdf_change_drv:
2720 0000BF6B E85EC1FFFF               <1>       call   change_current_drive
2721 0000BF70 726D                     <1>       jc     short msftdf_df_error_retn
2722                                    <1>
2723                                    <1> msftdf_check_destination_file:
2724                                    <1> msftdf_df_check_directory:
2725 0000BF72 BE[BD8C0100]             <1>       mov    esi, DestinationFile_Directory
2726 0000BF77 803E20                   <1>       cmp    byte [esi], 20h
2727 0000BF7A 760F                     <1>       jna    short msftdf_df_find_1
2728                                    <1>
2729                                    <1> msftdf_df_change_directory:
2730 0000BF7C FE05[23380100]           <1>       inc    byte [Restore_CDIR]
2731 0000BF82 30E4                     <1>       xor    ah, ah ; CD_COMMAND sign -> 0
2732 0000BF84 E8A3F0FFFF               <1>       call   change_current_directory
2733 0000BF89 7254                     <1>       jc     short msftdf_df_error_retn
2734                                    <1>
2735                                    <1> ;msftdf_df_change_prompt_dir_string:
2736                                    <1> ;     call   change_prompt_dir_string
2737                                    <1>
2738                                    <1> msftdf_df_find_1:
2739 0000BF8B BE[FE8C0100]             <1>         mov     esi, DestinationFile_Name
2740 0000BF90 803E20                   <1>       cmp    byte [esi], 20h
2741 0000BF93 7631                     <1>       jna    short msftdf_df_copy_sf_name
2742                                    <1>
2743                                    <1> msftdf_df_find_2:
2744 0000BF95 6631C0                   <1>       xor    ax, ax ; DestinationFile_AttributesMask -> any/zero
2745 0000BF98 E8D4D4FFFF               <1>       call   find_first_file
2746 0000BF9D 0F838D000000             <1>       jnc    msftdf_permission_denied_retn
2747                                    <1>
2748                                    <1> msftdf_df_check_error_code:
2749                                    <1>       ;cmp   eax, 2 ; File not found error
2750 0000BFA3 3C02                     <1>       cmp    al, 2
2751 0000BFA5 7537                     <1>       jne    short msftdf_df_stc_retn
2752                                    <1>
2753                                    <1> msftdf_df_check_fname:
2754                                    <1>       ; 15/10/2016
2755 0000BFA7 BE[FE8C0100]             <1>       mov    esi, DestinationFile_Name ; *
2756 0000BFAC E87ED8FFFF               <1>       call   check_filename
2757 0000BFB1 7307                     <1>       jnc    short msftdf_convert_df_direntry_name
2758                                    <1>       ; invalid file name chars !
2759 0000BFB3 B81A000000               <1>       mov    eax, ERR_INV_FILE_NAME  ; 26
2760 0000BFB8 EB24                     <1>       jmp    short msftdf_df_stc_retn
2761                                    <1>
2762                                    <1> msftdf_convert_df_direntry_name:
2763                                    <1>       ; mov  esi, DestinationFile_Name ; *
2764 0000BFBA BF[0E8D0100]             <1>       mov    edi, DestinationFile_DirEntry
2765 0000BFBF E8B3F5FFFF               <1>       call   convert_file_name
```

```
2766 0000BFC4 EB1A              <1>       jmp    short msftdf_restore_current_dir_1
2767                            <1>
2768                            <1> msftdf_df_copy_sf_name:
2769 0000BFC6 89F7              <1>       mov    edi, esi
2770 0000BFC8 57                <1>       push   edi
2771 0000BFC9 BE[7E8C0100]      <1>         mov     esi, SourceFile_Name
2772 0000BFCE B90C000000        <1>       mov    ecx, 12
2773                            <1> msftdf_df_copy_sf_name_loop:
2774 0000BFD3 AC                <1>       lodsb
2775 0000BFD4 AA                <1>         stosb
2776 0000BFD5 08C0              <1>       or     al, al
2777 0000BFD7 7402              <1>       jz     short msftdf_df_copy_sf_name_ok
2778 0000BFD9 E2F8              <1>         loop    msftdf_df_copy_sf_name_loop
2779                            <1> msftdf_df_copy_sf_name_ok:
2780 0000BFDB 5E                <1>       pop    esi
2781 0000BFDC EBB7              <1>       jmp    short msftdf_df_find_2
2782                            <1>
2783                            <1> msftdf_df_stc_retn:
2784 0000BFDE F9                <1>       stc
2785                            <1> msftdf_restore_cdir_failed:
2786                            <1> msftdf_df_error_retn:
2787 0000BFDF C3                <1>       retn
2788                            <1>
2789                            <1> msftdf_restore_current_dir_1:
2790 0000BFE0 803D[23380100]00  <1>       cmp    byte [Restore_CDIR], 0
2791 0000BFE7 760D              <1>       jna    short msftdf_sf_check_directory
2792 0000BFE9 8B35[408D0100]    <1>       mov    esi, [msftdf_drv_offset]
2793 0000BFEF E891C1FFFF        <1>       call   restore_current_directory
2794 0000BFF4 72E9              <1>       jc     short msftdf_restore_cdir_failed
2795                            <1>
2796                            <1> msftdf_sf_check_directory:
2797 0000BFF6 BE[3D8C0100]      <1>       mov    esi, SourceFile_Directory
2798 0000BFFB 803E20            <1>       cmp    byte [esi], 20h
2799 0000BFFE 760F              <1>       jna    short msftdf_sf_find
2800                            <1> msftdf_sf_change_directory:
2801 0000C000 FE05[23380100]    <1>       inc    byte [Restore_CDIR]
2802 0000C006 30E4              <1>       xor    ah, ah ; CD_COMMAND sign -> 0
2803 0000C008 E81FF0FFFF        <1>       call   change_current_directory
2804 0000C00D 7227              <1>       jc     short msftdf_return
2805                            <1>
2806                            <1> ;msftdf_sf_change_prompt_dir_string:
2807                            <1> ;     call   change_prompt_dir_string
2808                            <1>
2809                            <1> msftdf_sf_find:
2810 0000C00F BE[7E8C0100]      <1>         mov     esi, SourceFile_Name  ; Offset 66
2811 0000C014 66B80018          <1>       mov    ax, 1800h ; Only files
2812 0000C018 E854D4FFFF        <1>       call   find_first_file
2813 0000C01D 7217              <1>       jc     short msftdf_return
2814                            <1>
2815                            <1> msftdf_sf_ambgfn_check:
2816 0000C01F 6609D2            <1>       or     dx, dx ; Ambiguous filename chars used sign (DX>0)
2817 0000C022 7407              <1>       jz     short msftdf_sf_found
2818                            <1>
2819                            <1> msftdf_ambiguous_file_name_error:
2820 0000C024 B802000000        <1>       mov    eax, 2 ; File not found error
2821 0000C029 F9                <1>       stc
2822 0000C02A C3                <1>       retn
2823                            <1>
2824                            <1> msftdf_sf_found:
2825 0000C02B 80E31F            <1>       and    bl, 1Fh ; Attributes, D-V-S-H-R
2826 0000C02E 7416              <1>       jz     short msftdf_save_sf_structure
2827                            <1>
2828                            <1> msftdf_permission_denied_retn:
2829 0000C030 B805000000        <1>       mov    eax, 05h ; Access (Permission) denied !
2830 0000C035 F9                <1>       stc
2831                            <1> msftdf_rest_cdir_err_retn:
2832                            <1> msftdf_return:
2833 0000C036 C3                <1>       retn
2834                            <1>
2835                            <1> msftdf_phase_1_return:
2836 0000C037 31C0              <1>       xor    eax, eax
2837 0000C039 A2[3E8D0100]      <1>       mov    [move_cmd_phase], al ; 0
2838 0000C03E FEC0              <1>       inc    al ; mov al, 1
2839 0000C040 BB[8DC00000]      <1>       mov    ebx, msftdf_df2_check_directory
2840                            <1>       ;mov   edx, 0FFFFFFFFh
2841 0000C045 C3                <1>       retn
2842                            <1>
2843                            <1> msftdf_save_sf_structure:
2844 0000C046 BE[488B0100]      <1>       mov    esi, FindFile_DirEntry
2845 0000C04B BF[8E8C0100]      <1>       mov    edi, SourceFile_DirEntry
2846 0000C050 B908000000        <1>       mov    ecx, 8
2847 0000C055 F3A5              <1>       rep    movsd
2848                            <1>
2849                            <1> msftdf_df_copy_sf_parameters:
2850 0000C057 BE0B000000        <1>       mov    esi, 11
2851 0000C05C 89F7              <1>       mov    edi, esi
2852 0000C05E 81C6[8E8C0100]    <1>       add    esi, SourceFile_DirEntry
2853 0000C064 81C7[0E8D0100]    <1>       add    edi, DestinationFile_DirEntry
2854                            <1>       ;mov   ecx, 21
2855 0000C06A B115              <1>       mov    cl, 21
2856 0000C06C F3A4              <1>       rep    movsb
2857                            <1>
2858                            <1> msftdf_restore_current_dir_2:
2859 0000C06E 803D[23380100]00  <1>       cmp    byte [Restore_CDIR], 0
2860 0000C075 760D              <1>       jna    short msftdf_df2_check_move_cmd_phase
2861 0000C077 8B35[408D0100]    <1>       mov    esi, [msftdf_drv_offset]
2862 0000C07D E803C1FFFF        <1>       call   restore_current_directory
2863 0000C082 72B2              <1>       jc     short msftdf_rest_cdir_err_retn
2864                            <1>
2865                            <1> msftdf_df2_check_move_cmd_phase:
2866 0000C084 803D[3E8D0100]01  <1>       cmp    byte [move_cmd_phase], 1
2867 0000C08B 74AA              <1>       je     short msftdf_phase_1_return
2868                            <1>
2869                            <1> msftdf_df2_check_directory:
2870 0000C08D BE[BD8C0100]      <1>       mov    esi, DestinationFile_Directory
```

```
2871 0000C092 803E20              <1>       cmp   byte [esi], 20h
2872 0000C095 760F                <1>       jna   short msftdf_make_dfde_locate_ffe_on_directory
2873                              <1> msftdf_df2_change_directory:
2874 0000C097 FE05[23380100]      <1>       inc   byte [Restore_CDIR]
2875 0000C09D 30E4                <1>       xor   ah, ah ; CD_COMMAND sign -> 0
2876 0000C09F E888EFFFFF          <1>       call  change_current_directory
2877 0000C0A4 7290                <1>       jc    short msftdf_return
2878                              <1>
2879                              <1> ;msftdf_df2_change_prompt_dir_string:
2880                              <1> ;     call  change_prompt_dir_string
2881                              <1>
2882                              <1> msftdf_make_dfde_locate_ffe_on_directory:
2883                              <1>       ; Current directory fcluster <> Directory buffer cluster
2884                              <1>       ; Current directory will be reloaded by
2885                              <1>       ; 'locate_current_dir_file' procedure
2886                              <1>       ;
2887                              <1>       ;xor   ax, ax
2888 0000C0A6 31C0                <1>       xor   eax, eax
2889 0000C0A8 89C1                <1>       mov   ecx, eax
2890 0000C0AA 6649                <1>       dec   cx ; FFFFh
2891                              <1>             ; CX = FFFFh -> find first deleted or free entry
2892                              <1>             ; ESI would be ASCIIZ filename address if the call
2893                              <1>             ; would not be for first free or deleted dir entry
2894 0000C0AC E8CFF1FFFF          <1>       call  locate_current_dir_file
2895 0000C0B1 733F                <1>       jnc   msftdf_make_dfde_set_ff_dir_entry
2896                              <1>
2897                              <1>       ;cmp   eax, 2
2898 0000C0B3 3C02                <1>         cmp  al, 2
2899 0000C0B5 7537                <1>       jne   short msftdf_error_retn
2900                              <1>
2901                              <1> msftdf_add_new_dir_entry_check_fs:
2902 0000C0B7 8B35[408D0100]      <1>       mov   esi, [msftdf_drv_offset]
2903 0000C0BD A1[81890100]        <1>       mov   eax, [DirBuff_Cluster]
2904 0000C0C2 807E0300            <1>       cmp   byte [esi+LD_FATType], 0
2905 0000C0C6 7711                <1>       ja    short msftdf_add_new_subdir_cluster
2906                              <1>
2907                              <1> msftdf_add_new_fs_subdir_section:
2908                              <1>       ;CL=0, CH=E5h --> deleted entry, CH=0 --> free entry
2909                              <1>         ;xor cx, cx
2910 0000C0C8 30ED                <1>       xor   ch, ch ; cx = 0 --> add a new subdir section
2911 0000C0CA E8830C0000          <1>       call  add_new_fs_section
2912 0000C0CF 721E                <1>         jc  short msftdf_dsfde_error_retn
2913                              <1>       ;mov  [createfile_LastDirCluster], eax
2914                              <1>
2915 0000C0D1 E8A30E0000          <1>       call  load_FS_sub_directory
2916                              <1>       ;mov  ebx, Directory_Buffer
2917 0000C0D6 7318                <1>       jnc   short msftdf_add_new_fs_subdir_section_ok
2918 0000C0D8 C3                  <1>       retn
2919                              <1>
2920                              <1> msftdf_add_new_subdir_cluster:
2921 0000C0D9 E881150000          <1>       call  add_new_cluster
2922 0000C0DE 720F                <1>       jc    short msftdf_dsfde_error_retn
2923                              <1>
2924                              <1>       ;mov  [createfile_LastDirCluster], eax
2925                              <1>
2926 0000C0E0 E8570E0000          <1>       call  load_FAT_sub_directory
2927 0000C0E5 7309                <1>       jnc   short msftdf_add_new_subdir_cluster_ok
2928                              <1>       ; EBX = Directory buffer address
2929                              <1>
2930                              <1> msftdf_ansdc_update_parent_dir_lmdt:
2931                              <1> msftdf_make_dfde_err_upd_pdir_lmdt:
2932 0000C0E7 50                  <1>       push  eax
2933 0000C0E8 E854FAFFFF          <1>       call  update_parent_dir_lmdt
2934 0000C0ED 58                  <1>       pop   eax
2935                              <1>
2936                              <1> msftdf_error_retn:
2937 0000C0EE F9                  <1>       stc
2938                              <1> msftdf_dsfde_restore_cdir_failed:
2939                              <1> msftdf_dsfde_error_retn:
2940 0000C0EF C3                  <1>       retn
2941                              <1>
2942                              <1> msftdf_add_new_fs_subdir_section_ok:
2943                              <1> msftdf_add_new_subdir_cluster_ok:
2944 0000C0F0 89DF                <1>       mov   edi, ebx ; Directory buffer address
2945                              <1>
2946                              <1> msftdf_make_dfde_set_ff_dir_entry:
2947 0000C0F2 8B15[50820100]      <1>       mov   edx, [Current_Dir_FCluster]
2948 0000C0F8 8915[A48D0100]      <1>       mov   [createfile_FFCluster], edx
2949                              <1>       ; EDI = Directory entry offset
2950 0000C0FE BE[0E8D0100]        <1>       mov   esi, DestinationFile_DirEntry
2951 0000C103 B908000000          <1>       mov   ecx, 8
2952 0000C108 F3A5                <1>       rep   movsd
2953                              <1>
2954 0000C10A C605[7C890100]02    <1>       mov   byte [DirBuff_ValidData], 2
2955 0000C111 E890F9FFFF          <1>       call  save_directory_buffer
2956 0000C116 72CF                <1>       jc    short msftdf_make_dfde_err_upd_pdir_lmdt
2957                              <1>
2958                              <1> msftdf_make_dfde_update_pdir_lmdt:
2959 0000C118 E824FAFFFF          <1>       call  update_parent_dir_lmdt
2960                              <1>
2961                              <1> msftdf_dsfde_restore_current_dir_1:
2962 0000C11D 803D[23380100]00    <1>       cmp   byte [Restore_CDIR], 0
2963 0000C124 760D                <1>       jna   short msftdf_dsfde_check_directory
2964 0000C126 8B35[408D0100]      <1>       mov   esi, [msftdf_drv_offset]
2965 0000C12C E854C0FFFF          <1>       call  restore_current_directory
2966 0000C131 72BC                <1>       jc    short msftdf_dsfde_restore_cdir_failed
2967                              <1>
2968                              <1> msftdf_dsfde_check_directory:
2969 0000C133 BE[3D8C0100]        <1>       mov   esi, SourceFile_Directory
2970 0000C138 803E20              <1>       cmp   byte [esi], 20h
2971 0000C13B 760F                <1>       jna   short msftdf_dsfde_find_file
2972                              <1>
2973                              <1> msftdf_dsfde_change_directory:
2974 0000C13D FE05[23380100]      <1>       inc   byte [Restore_CDIR]
2975 0000C143 28E4                <1>       sub   ah, ah ; CD_COMMAND sign -> 0
```

```
2976 0000C145 E8E2EEFFFF          <1>       call   change_current_directory
2977 0000C14A 72A3                <1>       jc     short msftdf_dsfde_error_retn
2978                              <1>
2979                              <1> ;msftdf_dsfde_sf_change_prompt_dir_string:
2980                              <1> ;     call   change_prompt_dir_string
2981                              <1>
2982                              <1> msftdf_dsfde_find_file:
2983 0000C14C BE[7E8C0100]        <1>       mov    esi, SourceFile_Name  ; Offset 66
2984 0000C151 668B460E            <1>       mov    ax, [esi+14] ; 80 -> SourceFile_AttributesMask
2985 0000C155 E817D3FFFF          <1>       call   find_first_file
2986 0000C15A 7293                <1>       jc     short msftdf_dsfde_error_retn
2987                              <1>
2988                              <1> msftdf_dsfde_delete_direntry:
2989 0000C15C 8B35[408D0100]      <1>       mov    esi, [msftdf_drv_offset]
2990                              <1>
2991 0000C162 807E0300            <1>       cmp    byte [esi+LD_FATType], 0
2992 0000C166 770A                <1>       ja     short msftdf_delete_FAT_direntry
2993                              <1>
2994 0000C168 30DB                <1>       xor    bl, bl
2995                              <1>       ; BL = 0 -> File
2996                              <1>       ; EDI -> Directory buffer entry offset/address
2997 0000C16A E8E40B0000          <1>       call   delete_fs_directory_entry
2998 0000C16F 7315                <1>       jnc    short msftdf_dsfde_restore_current_dir_2
2999 0000C171 C3                  <1>       retn
3000                              <1>
3001                              <1> msftdf_delete_FAT_direntry:
3002 0000C172 8A1D[458B0100]      <1>       mov    bl, [FindFile_LongNameEntryLength]
3003 0000C178 668B0D[708B0100]    <1>       mov    cx, [FindFile_DirEntryNumber]
3004                              <1>       ; ESI = Logical DOS drive description table address
3005                              <1>       ; EDI = Directory buffer entry offset/address
3006 0000C17F E89AFCFFFF          <1>       call   delete_directory_entry
3007 0000C184 721C                <1>       jc     short msftdf_retn
3008                              <1>
3009                              <1> msftdf_dsfde_restore_current_dir_2:
3010 0000C186 803D[23380100]00    <1>       cmp    byte [Restore_CDIR], 0
3011 0000C18D 7607                <1>       jna    short msftdf_new_dir_fcluster_retn
3012                              <1>       ;mov   esi, [msftdf_drv_offset]
3013 0000C18F E8F1BFFFFF          <1>       call   restore_current_directory
3014 0000C194 720C                <1>       jc     short msftdf_retn
3015                              <1>
3016                              <1> msftdf_new_dir_fcluster_retn:
3017 0000C196 31C9                <1>       xor    ecx, ecx
3018 0000C198 A1[A48D0100]        <1>       mov    eax, [createfile_FFCluster]
3019 0000C19D BB[BC8C0100]        <1>       mov    ebx, DestinationFile_Drv
3020                              <1>
3021                              <1> msftdf_retn:
3022 0000C1A2 C3                  <1>       retn
3023                              <1>
3024                              <1>
3025                              <1> copy_source_file_to_destination_file:
3026                              <1>       ; 17/10/2016
3027                              <1>       ; 16/10/2016
3028                              <1>       ; 15/10/2016
3029                              <1>       ; 30/03/2016, 31/03/2016
3030                              <1>       ; 24/03/2016, 25/03/2016, 28/03/2016
3031                              <1>       ; 21/03/2016, 22/03/2016, 23/03/2016
3032                              <1>       ; 16/03/2016, 17/03/2016, 18/03/2016
3033                              <1>       ; 15/03/2016 (TRDOS 386 = TRDOS v2.0)
3034                              <1>       ; 02/09/2011 (FILE.ASM 'copy_source_file_to_destination_file')
3035                              <1>       ; 01/08/2010 - 18/05/2011
3036                              <1>       ;
3037                              <1>       ;   Command Interpreter phase 1 enter ->
3038                              <1>       ;           AL = 1 -> Caller is command interpreter
3039                              <1>       ;           AL = 2 -> The second call, re-enter/continue
3040                              <1>       ;   Phase 1 -> Check source file
3041                              <1>       ;              'found' is required
3042                              <1>       ;   Phase 2 -> Check destination file,
3043                              <1>       ;              save 'found' or 'not found' status
3044                              <1>       ;              'permission denied' error will be return
3045                              <1>       ;              if attributes have not for ordinary file
3046                              <1>       ;              without readonly attribute
3047                              <1>       ;   Command Interpreter phase 1 return ->
3048                              <1>       ;              DH = Source file attributes
3049                              <1>       ;              DL = Destination file found status
3050                              <1>       ;              EAX = 0
3051                              <1>       ;   Command Interpreter phase 2 enter ->
3052                              <1>       ;              AL = 2 -> Continue from the last position
3053                              <1>       ;              AH =
3054                              <1>       ;   Phase 3 -> Load source file or use read/write cluster method
3055                              <1>       ;   Phase 4 -> Create destination file if it is not found
3056                              <1>       ;   Phase 5 -> Open destination file
3057                              <1>       ;   Phase 6 -> Read from source and write to destination
3058                              <1>       ;   Phase 7 -> Unload source file, if it is loaded at memory
3059                              <1>       ;       cf = 1 causes to return before the phase 7
3060                              <1>       ;              but loaded file will be unloaded
3061                              <1>       ;             (allocated memory block will be deallocated)
3062                              <1>       ;
3063                              <1>       ; INPUT ->
3064                              <1>       ;      ESI = Source File Pathname (Asciiz)
3065                              <1>       ;        EDI = Destination File Pathname (Asciiz)
3066                              <1>       ;        AL = 0 --> Interrupt (System call)
3067                              <1>       ;        AL > 0 --> Command Interpreter (Question)
3068                              <1>       ;        AL = 1 --> Question Phase
3069                              <1>       ;        AL = 2 --> Progress Phase
3070                              <1>       ;
3071                              <1>       ; OUTPUT ->
3072                              <1>       ;     cf = 0 -> OK
3073                              <1>       ;     EAX = Destination file first cluster
3074                              <1>       ;
3075                              <1>       ;        CL > 0 if there is file reading error before EOF
3076                              <1>       ;              (incomplete copy)
3077                              <1>       ;        CH > 0 if file is (full) loaded at memory
3078                              <1>       ;
3079                              <1>       ;     cf = 1 -> Error code in AL (EAX)
3080                              <1>       ;
```

```
                        <1>       ; (EBX, ECX, ESI, EDI register contents will be changed)
                        <1>
                        <1>
0000C1A3 3C02           <1>       cmp    al, 2
0000C1A5 0F845A020000   <1>       je     csftdf2_check_cdrv
                        <1>
                        <1> ; Phase 1
                        <1>
0000C1AB A2[648D0100]   <1>       mov    byte [copy_cmd_phase], al
                        <1>
0000C1B0 57             <1>       push   edi ; *
                        <1>
                        <1> csftdf_parse_sf_path:
0000C1B1 BF[3C8C0100]   <1>       mov    edi, SourceFile_Drv
0000C1B6 E887F4FFFF     <1>       call   parse_path_name
0000C1BB 721C           <1>       jc     short csftdf_parse_sf_path_failed
                        <1>
                        <1> csftdf_parse_df_path:
0000C1BD 5E             <1>       pop    esi ; * (pushed edi)
                        <1>
                        <1> csftdf_sf_check_filename_exists:
0000C1BE 803D[7E8C0100]21 <1>     cmp    byte [SourceFile_Name], 21h
0000C1C5 7215           <1>       jb     short csftdf_sf_file_not_found_error
                        <1>
0000C1C7 BF[BC8C0100]   <1>       mov    edi, DestinationFile_Drv
0000C1CC E871F4FFFF     <1>       call   parse_path_name
0000C1D1 7310           <1>       jnc    short csftdf_check_sf_cdrv
                        <1>
0000C1D3 3C01           <1>       cmp    al, 1 ; File or directory name is not existing
0000C1D5 760C           <1>       jna    short csftdf_check_sf_cdrv
                        <1>
                        <1> csftdf_parse_df_path_failed:
0000C1D7 F9             <1>       stc
                        <1> csftdf_sf_error_retn:
0000C1D8 C3             <1>       retn
                        <1>
                        <1> csftdf_parse_sf_path_failed:
0000C1D9 5F             <1>       pop    edi ; *
0000C1DA EBFC           <1>       jmp    short csftdf_sf_error_retn
                        <1>
                        <1> csftdf_sf_file_not_found_error:
0000C1DC B802000000     <1>       mov    eax, 2 ; File not found
0000C1E1 EBF5           <1>       jmp    short csftdf_sf_error_retn
                        <1>
                        <1> csftdf_check_sf_cdrv:
0000C1E3 8A3D[56820100] <1>       mov    bh, [Current_Drv]
                        <1>
0000C1E9 883D[678D0100] <1>       mov    [csftdf_cdrv], bh ; 23/03/2016
                        <1>
0000C1EF 8A15[3C8C0100] <1>       mov    dl, [SourceFile_Drv]
0000C1F5 38FA           <1>       cmp    dl, bh ; byte [Current_Drv]
0000C1F7 7407           <1>       je     short csftdf_sf_check_directory
                        <1>
0000C1F9 E8D0BEFFFF     <1>       call   change_current_drive
0000C1FE 72D8           <1>       jc     short csftdf_sf_error_retn
                        <1>
                        <1> csftdf_sf_check_directory:
0000C200 BE[3D8C0100]   <1>       mov    esi, SourceFile_Directory
0000C205 803E20         <1>       cmp    byte [esi], 20h
0000C208 760F           <1>       jna    short csftdf_find_sf
                        <1>
                        <1> csftdf_sf_change_directory:
0000C20A FE05[23380100] <1>       inc    byte [Restore_CDIR]
0000C210 30E4           <1>       xor    ah, ah ; CD_COMMAND sign -> 0
0000C212 E815EEFFFF     <1>       call   change_current_directory
0000C217 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:
0000C219 BE[7E8C0100]   <1>       mov    esi, SourceFile_Name
0000C21E 66B80018       <1>       mov    ax, 1800h ; Except volume label and dirs
0000C222 E84AD2FFFF     <1>       call   find_first_file
0000C227 72AF           <1>       jc     short csftdf_sf_error_retn
                        <1>
                        <1> csftdf_sf_ambgfn_check:
0000C229 6621D2         <1>       and    dx, dx ; Ambiguous filename chars used sign (DX>0)
0000C22C 7407           <1>       jz     short csftdf_sf_found
                        <1>
                        <1> csftdf_ambiguous_file_name_error:
0000C22E B802000000     <1>       mov    eax, 2 ; File not found error
0000C233 F9             <1>       stc
0000C234 C3             <1>       retn
                        <1>
                        <1> csftdf_sf_found:
0000C235 A3[688D0100]   <1>       mov    [csftdf_filesize], eax
                        <1>
0000C23A 09C0           <1>       or     eax, eax
0000C23C 7507           <1>       jnz    short csftdf_set_source_file_direntry
                        <1>
                        <1> csftdf_sf_file_size_zero:
0000C23E B814000000     <1>       mov    eax, 20 ; TRDOS zero length (file size) error
0000C243 F9             <1>       stc
0000C244 C3             <1>       retn
                        <1>
                        <1> csftdf_set_source_file_direntry:
0000C245 BE[488B0100]   <1>       mov    esi, FindFile_DirEntry
0000C24A BF[8E8C0100]   <1>       mov    edi, SourceFile_DirEntry
0000C24F B908000000     <1>       mov    ecx, 8
0000C254 F3A5           <1>       rep    movsd
                        <1>
                        <1> csftdf_sf_restore_cdrv:
                        <1>       ; 22/03/2016
0000C256 8A15[678D0100] <1>       mov    dl, [csftdf_cdrv]
```

```
3186 0000C25C 3A15[56820100]      <1>       cmp    dl, [Current_Drv]
3187 0000C262 7407                <1>       je     short csftdf_sf_restore_cdir
3188 0000C264 E865BEFFFF          <1>       call   change_current_drive
3189 0000C269 724F                <1>       jc     short csftdf_df_error_retn ; 30/03/2016
3190                              <1>
3191                              <1> csftdf_sf_restore_cdir:
3192 0000C26B 803D[23380100]00    <1>       cmp    byte [Restore_CDIR], 0
3193 0000C272 7612                <1>       jna    short csftdf_df_check_filename_exists
3194 0000C274 29C0                <1>       sub    eax, eax
3195 0000C276 BE00010900          <1>       mov    esi, Logical_DOSDisks
3196 0000C27B 88D4                <1>       mov    ah, dl ; byte [csftdf_cdrv]
3197 0000C27D 01C6                <1>       add    esi, eax
3198 0000C27F E801BFFFFF          <1>       call   restore_current_directory
3199 0000C284 7234                <1>       jc     short csftdf_df_error_retn
3200                              <1>
3201                              <1> csftdf_df_check_filename_exists:
3202 0000C286 803D[FE8C0100]20    <1>       cmp    byte [DestinationFile_Name], 20h
3203 0000C28D 7716                <1>       ja     short csftdf_check_df_cdrv
3204                              <1>
3205                              <1> csftdf_copy_sf_name:
3206 0000C28F BF[FE8C0100]        <1>       mov    edi, DestinationFile_Name
3207 0000C294 BE[7E8C0100]        <1>       mov    esi, SourceFile_Name
3208 0000C299 B10C                <1>       mov    cl, 12
3209                              <1>
3210                              <1> csftdf_df_copy_sf_name_loop:
3211 0000C29B AC                  <1>       lodsb
3212 0000C29C AA                  <1>       stosb
3213 0000C29D 08C0                <1>       or     al, al
3214 0000C29F 7404                <1>       jz     short csftdf_check_df_cdrv
3215 0000C2A1 FEC9                <1>       dec    cl
3216 0000C2A3 75F6                <1>       jnz    csftdf_df_copy_sf_name_loop
3217                              <1>
3218                              <1> csftdf_check_df_cdrv:
3219 0000C2A5 8A15[BC8C0100]      <1>       mov    dl, [DestinationFile_Drv]
3220 0000C2AB 3A15[56820100]      <1>       cmp    dl, [Current_Drv]
3221 0000C2B1 7408                <1>       je     short csftdf_df_check_directory
3222                              <1>
3223 0000C2B3 E816BEFFFF          <1>       call   change_current_drive
3224 0000C2B8 7301                <1>       jnc    short csftdf_df_check_directory
3225                              <1>
3226                              <1> csftdf_df_error_retn:
3227 0000C2BA C3                  <1>       retn
3228                              <1>
3229                              <1> csftdf_df_check_directory:
3230 0000C2BB BE[BD8C0100]        <1>       mov    esi, DestinationFile_Directory
3231 0000C2C0 803E20              <1>         cmp     byte [esi], 20h
3232 0000C2C3 760F                <1>       jna    short csftdf_find_df
3233                              <1>
3234                              <1> csftdf_df_change_directory:
3235 0000C2C5 FE05[23380100]      <1>       inc    byte [Restore_CDIR]
3236 0000C2CB 28E4                <1>       sub    ah, ah ; CD_COMMAND sign -> 0
3237 0000C2CD E85AEDFFFF          <1>       call   change_current_directory
3238 0000C2D2 72E6                <1>       jc     short csftdf_df_error_retn
3239                              <1>
3240                              <1> ;csftdf_df_change_prompt_dir_string:
3241                              <1> ;     call   change_prompt_dir_string
3242                              <1>
3243                              <1> csftdf_find_df:
3244                              <1>       ; 23/03/2016
3245 0000C2D4 29DB                <1>       sub    ebx, ebx
3246 0000C2D6 8A3D[BC8C0100]      <1>       mov    bh,  [DestinationFile_Drv]
3247 0000C2DC 81C300010900        <1>       add    ebx, Logical_DOSDisks
3248 0000C2E2 891D[948D0100]      <1>       mov    [csftdf_df_drv_dt], ebx
3249                              <1>
3250 0000C2E8 BE[FE8C0100]        <1>       mov    esi, DestinationFile_Name
3251 0000C2ED 6631C0              <1>       xor    ax, ax
3252                              <1>              ; DestinationFile_AttributesMask -> any/zero
3253 0000C2F0 E87CD1FFFF          <1>       call   find_first_file
3254 0000C2F5 7218                <1>       jc     short csftdf_df_check_error_code
3255                              <1>
3256                              <1> csftdf_df_ambgfn_check:
3257 0000C2F7 6609D2              <1>       or     dx, dx ; Ambiguous filename chars used sign (DX>0)
3258 0000C2FA 752A                <1>       jnz    short csftdf_df_error_inv_fname
3259                              <1>
3260                              <1> csftdf_df_found:
3261 0000C2FC C605[668D0100]01    <1>       mov    byte [DestinationFileFound], 1
3262                              <1>       ; 17/10/2016 (cl -> bl)
3263 0000C303 80E31F              <1>       and    bl, 1Fh ; Attributes, D-V-S-H-R
3264 0000C306 745F                <1>       jz     short csftdf_df_save_first_cluster
3265                              <1>
3266                              <1> csftdf_df_permission_denied_retn:
3267 0000C308 B805000000          <1>       mov    eax, 05h ; Access/Permission denied.
3268                              <1> csftdf_df_error_stc_retn:
3269 0000C30D F9                  <1>       stc
3270 0000C30E C3                  <1>       retn
3271                              <1>
3272                              <1> csftdf_df_check_error_code:
3273                              <1>       ;cmp   eax, 2
3274 0000C30F 3C02                <1>       cmp    al, 2
3275 0000C311 75FA                <1>       jne    short csftdf_df_error_stc_retn
3276                              <1>
3277 0000C313 C605[668D0100]00    <1>       mov    byte [DestinationFileFound], 0
3278                              <1>
3279                              <1>       ; 15/10/2016
3280 0000C31A BE[388B0100]        <1>       mov    esi, FindFile_Name ; *
3281 0000C31F E80BD5FFFF          <1>       call   check_filename
3282 0000C324 7307                <1>       jnc    short csftdf_df_valid_fname
3283                              <1> csftdf_df_error_inv_fname: ; 'invalid file name !'
3284 0000C326 B81A000000          <1>       mov    eax, ERR_INV_FILE_NAME  ; 26
3285 0000C32B F9                  <1>       stc
3286 0000C32C C3                  <1>       retn
3287                              <1>
3288                              <1> csftdf_df_valid_fname:
3289                              <1>       ; 21/03/2016
3290                              <1>       ; (Capitalized file name)
```

```
3291                                    <1>       ;mov   esi, FindFile_Name ; * ; 15/10/2016
3292 0000C32D BF[FE8C0100]              <1>       mov    edi, DestinationFile_Name
3293 0000C332 A5                        <1>       movsd
3294 0000C333 A5                        <1>       movsd
3295 0000C334 A5                        <1>       movsd
3296                                    <1>       ;movsb
3297                                    <1>
3298                                    <1> csftdf_check_disk_free_size_0:
3299 0000C335 A1[AA8C0100]              <1>       mov    eax, [SourceFile_DirEntry+DirEntry_FileSize]
3300                                    <1>
3301                                    <1> csftdf_check_disk_free_size_1:
3302                                    <1>       ;sub   ebx, ebx
3303                                    <1>       ;mov   esi, Logical_DOSDisks
3304                                    <1>       ;mov   bh,  [DestinationFile_Drv]
3305                                    <1>       ;add   esi, ebx
3306                                    <1>
3307 0000C33A 8B35[948D0100]            <1>       mov    esi, [csftdf_df_drv_dt] ; 23/03/2016
3308                                    <1>
3309 0000C340 0FB74E11                 <1>       movzx  ecx, word [esi+LD_BPB+BytesPerSec] ; 17, LD_BPB + 0Bh
3310 0000C344 01C8                     <1>       add    eax, ecx
3311 0000C346 48                       <1>       dec    eax  ; file size (additional bytes) + 511 (round up)
3312                                    <1> csftdf_check_disk_free_size_3: ; 16/03/2016
3313 0000C347 29D2                     <1>       sub    edx, edx
3314 0000C349 F7F1                     <1>       div    ecx ; bytes per sector
3315                                    <1>
3316                                    <1> csftdf_check_disk_free_size:
3317 0000C34B 3B4674                   <1>       cmp    eax, [esi+LD_FreeSectors]
3318 0000C34E 0F8294000000             <1>         jb     csftdf_check_disk_free_size_ok
3319 0000C354 770A                     <1>       ja     short csftdf_df_insufficient_disk_space
3320                                    <1>
3321 0000C356 807E0300                 <1>       cmp    byte [esi+LD_FATType], 0 ; FS needs FDT sector also.
3322 0000C35A 0F8788000000             <1>         ja     csftdf_check_disk_free_size_ok
3323                                    <1>
3324                                    <1> csftdf_df_insufficient_disk_space:
3325 0000C360 B827000000               <1>       mov    eax, 27h ; insufficient disk space
3326 0000C365 EBA6                     <1>       jmp    short csftdf_df_error_stc_retn
3327                                    <1>
3328                                    <1> csftdf_df_save_first_cluster:
3329                                    <1>       ; ESI = FindFile_DirEntry (for the old destination file)
3330                                    <1>       ; EAX = Old destination file size
3331                                    <1>       ; 24/03/2016
3332                                    <1>       ; EDI = Directory entry address (within Dir Buffer boundaries)
3333 0000C367 81EF00000800              <1>       sub    edi, Directory_Buffer  ; (<65536)
3334 0000C36D 66C1EF05                 <1>       shr    di, 5 ; Convert entry offset to entry index/number
3335 0000C371 66893D[368D0100]         <1>       mov    [DestinationFile_DirEntryNumber], di ; (<2048)
3336                                    <1>
3337                                    <1> csftdf_df_check_sf_df_fcluster:
3338 0000C378 668B5614                 <1>       mov    dx, [esi+DirEntry_FstClusHI]
3339 0000C37C C1E210                   <1>       shl    edx, 16
3340 0000C37F 668B561A                 <1>       mov    dx, [esi+DirEntry_FstClusLO]
3341 0000C383 8915[788D0100]           <1>       mov    [csftdf_df_cluster], edx
3342                                    <1> csftdf_df_check_sf_df_fcluster_1:
3343 0000C389 668B15[A28C0100]         <1>       mov    dx, [SourceFile_DirEntry+DirEntry_FstClusHI]
3344 0000C390 C1E210                   <1>       shl    edx, 16
3345 0000C393 668B15[A88C0100]         <1>       mov    dx, [SourceFile_DirEntry+DirEntry_FstClusLO]
3346 0000C39A 3B15[788D0100]           <1>       cmp    edx, [csftdf_df_cluster]
3347 0000C3A0 7512                     <1>       jne    short csftdf_df_check_sf_df_fcluster_ok
3348                                    <1> csftdf_df_check_sf_df_drv:
3349 0000C3A2 8A15[3C8C0100]           <1>       mov    dl, [SourceFile_Drv]
3350 0000C3A8 3A15[BC8C0100]           <1>       cmp    dl, [DestinationFile_Drv]
3351 0000C3AE 7504                     <1>       jne    short csftdf_df_check_sf_df_fcluster_ok
3352                                    <1>
3353                                    <1>       ; source and destination files are same !
3354                                    <1>       ; (they have same first cluster value on same logical disk)
3355                                    <1>
3356 0000C3B0 31C0                     <1>       xor    eax, eax ; mov eax, 0 -> Bad command or file name !
3357 0000C3B2 F9                       <1>       stc
3358 0000C3B3 C3                       <1>       retn
3359                                    <1>
3360                                    <1> csftdf_df_check_sf_df_fcluster_ok:
3361                                    <1> csftdf_df_move_findfile_struct:
3362                                    <1>       ; mov  esi, FindFile_DirEntry
3363 0000C3B4 BF[0E8D0100]              <1>       mov    edi, DestinationFile_DirEntry
3364 0000C3B9 B908000000               <1>       mov    ecx, 8
3365 0000C3BE F3A5                     <1>       rep    movsd
3366                                    <1>
3367                                    <1> csftdf_check_disk_free_size_2:
3368 0000C3C0 89C2                     <1>       mov    edx, eax ; Old destination file size
3369                                    <1>
3370                                    <1>       ;mov   eax, [SourceFile_DirEntry+DirEntry_FileSize]
3371 0000C3C2 A1[688D0100]             <1>       mov    eax, [csftdf_filesize] ; 23/03/2016
3372                                    <1>
3373                                    <1>       ;;sub  ecx, ecx ; 0
3374                                    <1>       ;mov   esi, Logical_DOSDisks
3375                                    <1>       ;mov   ch,  [DestinationFile_Drv]
3376                                    <1>       ;add   esi, ecx
3377                                    <1>       ;
3378                                    <1>       ;mov   [csftdf_df_drv_dt], esi
3379                                    <1>
3380 0000C3C7 8B35[948D0100]           <1>       mov    esi, [csftdf_df_drv_dt] ; 23/03/2016
3381                                    <1>
3382 0000C3CD 668B4E11                 <1>       mov    cx, [esi+LD_BPB+BytesPerSec] ; 17, LD_BPB + 0Bh
3383 0000C3D1 01CA                     <1>       add    edx, ecx ; + 512
3384 0000C3D3 01C8                     <1>       add    eax, ecx ; + 512
3385 0000C3D5 4A                       <1>       dec    edx ; old file size + 511 (round up)
3386 0000C3D6 48                       <1>       dec    eax ; new file size + 511 (round up)
3387 0000C3D7 F7D9                     <1>       neg    ecx ; -512 ; 0FFFFFE00h
3388 0000C3D9 21CA                     <1>       and    edx, ecx ; = old sector count * 512
3389 0000C3DB 21C8                     <1>       and    eax, ecx ; = new sector count * 512
3390                                    <1>
3391 0000C3DD 29D0                     <1>       sub    eax, edx ; new file size - old file size (on disk)
3392 0000C3DF 7607                     <1>       jna    short csftdf_check_disk_free_size_ok
3393                                    <1>
3394 0000C3E1 F7D9                     <1>       neg    ecx ; 512 (bytes per sector) ; 200h
3395                                    <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)
0000C3E3 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:
0000C3E8 A0[648D0100]           <1>       mov     al, [copy_cmd_phase]
0000C3ED 3C01                   <1>       cmp     al, 1
0000C3EF 7514                   <1>       jne     short csftdf2_check_cdrv
                                <1>
0000C3F1 31C0                   <1>       xor     eax, eax
0000C3F3 A2[648D0100]           <1>       mov     [copy_cmd_phase], al ; 0
                                <1>
0000C3F8 8A15[668D0100]         <1>       mov     dl, [DestinationFileFound]
0000C3FE 8A35[998C0100]         <1>       mov     dh, [SourceFile_DirEntry+11] ; Attributes
                                <1>
                                <1> csftdf_return:
0000C404 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
0000C405 803D[668D0100]00       <1>       cmp     byte [DestinationFileFound], 0
0000C40C 7739                   <1>       ja      short csftdf2_set_sf_percentage
                                <1>
                                <1> csftdf2_create_file:
0000C40E BE[FE8C0100]           <1>       mov     esi, DestinationFile_Name
0000C413 A1[688D0100]           <1>       mov     eax, [csftdf_filesize]
0000C418 30C9                   <1>       xor     cl, cl ; 0
                                <1>
0000C41A 31DB                   <1>       xor     ebx, ebx ; 0
0000C41C 4B                     <1>       dec     ebx ; 0FFFFFFFFh
                                <1>
                                <1>       ; INPUT ->
                                <1>       ;       EAX -> File Size
                                <1>       ;       ESI = ASCIIZ File name
                                <1>       ;        CL = File attributes
                                <1>       ;       EBX = FFFFFFFFh -> empty file sign for FAT fs
                                <1>       ;       EBX <> FFFFFFFFh -> use file size for FAT fs
                                <1>       ;
                                <1>       ; OUTPUT ->
                                <1>       ;       EAX = New file's first cluster
                                <1>       ;       ESI = Logical Dos Drv Descr. Table Addr.
                                <1>       ;       EBX = CreateFile_Size address
                                <1>       ;       ECX = Sectors per cluster (<256)
                                <1>       ;       EDX = Directory Entry Index/Number (<65536)
                                <1>       ;
                                <1>       ;       cf = 1 -> error code in AL (EAX)
                                <1>
0000C41D E8EC050000             <1>       call    create_file
                                <1>       ;pop    esi
0000C422 0F82A3050000           <1>         jc      csftdf2_rw_error
                                <1>
                                <1> csftdf2_create_file_OK:
0000C428 A3[788D0100]           <1>       mov     [csftdf_df_cluster], eax
                                <1>
                                <1>       ; 24/03/2016
0000C42D 668915[368D0100]       <1>       mov     [DestinationFile_DirEntryNumber], dx
                                <1>
                                <1>       ; 21/03/2016
0000C434 BE00000800             <1>       mov     esi, Directory_Buffer
0000C439 C1E205                 <1>       shl     edx, 5 ; 32 * index number
0000C43C 01D6                   <1>       add     esi, edx
0000C43E BF[0E8D0100]           <1>       mov     edi, DestinationFile_DirEntry
0000C443 B108                   <1>       mov     cl, 8 ; 32 bytes
0000C445 F3A5                   <1>       rep     movsd
                                <1>
                                <1> csftdf2_set_sf_percentage:
                                <1>       ; 17/03/2016
0000C447 31C0                   <1>       xor     eax, eax
0000C449 A2[8C8D0100]           <1>       mov     [csftdf_percentage], al ; 0, reset
                                <1>
0000C44E A3[848D0100]           <1>       mov     [csftdf_sf_rbytes], eax ; 0, reset
0000C453 A3[888D0100]           <1>       mov     [csftdf_df_wbytes], eax ; 0, reset
                                <1>
0000C458 8A25[3C8C0100]         <1>       mov     ah, [SourceFile_Drv]
```

```
3501 0000C45E BE00010900          <1>       mov   esi, Logical_DOSDisks
3502 0000C463 01C6                <1>       add   esi, eax
3503                              <1>
3504 0000C465 8935[908D0100]      <1>       mov   [csftdf_sf_drv_dt], esi ; 23/03/2016
3505                              <1>
3506 0000C46B 668B15[A28C0100]    <1>       mov   dx, [SourceFile_DirEntry+DirEntry_FstClusHI]
3507 0000C472 C1E210              <1>       shl   edx, 16
3508 0000C475 668B15[A88C0100]    <1>       mov   dx, [SourceFile_DirEntry+DirEntry_FstClusLO]
3509 0000C47C 8915[748D0100]      <1>       mov   [csftdf_sf_cluster], edx
3510                              <1>
3511                              <1>       ; 16/03/2016
3512                              <1>       ; Note: Singlix FS boot sector parameters (for cluster
3513                              <1>       ;     related calculations) has same offset
3514                              <1>       ;     values from LD_BPB as in FAT file system.
3515                              <1>       ;     [esi+LD_BPB+SecPerClust] is 1 for Singlix FS.
3516                              <1>       ;
3517 0000C482 0FB64E13            <1>       movzx ecx, byte [esi+LD_BPB+SecPerClust]
3518 0000C486 880D[BA8C0100]      <1>       mov   [SourceFile_SecPerClust], cl
3519                              <1>
3520                              <1>       ; 17/03/2016
3521 0000C48C 386E03              <1>       cmp   [esi+LD_FATType], ch ; 0
3522 0000C48F 7707                <1>       ja    short csftdf2_set_sf_percent_rsize1
3523                              <1>
3524 0000C491 B800000100          <1>       mov   eax, 65536 ; read/write buffer size for Singlix FS
3525 0000C496 EB06                <1>       jmp   short csftdf2_set_sf_percent_rsize2
3526                              <1>
3527                              <1> csftdf2_set_sf_percent_rsize1:
3528 0000C498 668B4611            <1>       mov   ax, [esi+LD_BPB+BytesPerSec]
3529 0000C49C F7E1                <1>       mul   ecx
3530                              <1>       ;sub  edx, edx
3531                              <1> csftdf2_set_sf_percent_rsize2:
3532 0000C49E A3[7C8D0100]        <1>       mov   [csftdf_r_size], eax
3533                              <1>
3534                              <1> csftdf2_set_df_percentage:
3535                              <1>       ;sub  eax, eax
3536                              <1>       ;mov  ah, [DestinationFile_Drv]
3537                              <1>       ;mov  edi, Logical_DOSDisks
3538                              <1>       ;add  edi, eax
3539                              <1>       ;mov  [csftdf_df_drv_dt], edi ; 17/03/2016
3540                              <1>
3541 0000C4A3 8B3D[948D0100]      <1>       mov   edi, [csftdf_df_drv_dt] ; 23/03/2016
3542                              <1>
3543                              <1>       ; 16/03/2016
3544                              <1>       ; Note: Singlix FS boot sector parameters (for cluster
3545                              <1>       ;     related calculations) has same offset
3546                              <1>       ;     values from LD_BPB as in FAT file system.
3547                              <1>       ;     [edi+LD_BPB+SecPerClust] is 1 for Singlix FS.
3548                              <1>       ;
3549                              <1>       ;movzx ecx, byte [edi+LD_BPB+SecPerClust]
3550 0000C4A9 8A4F13              <1>       mov   cl, [edi+LD_BPB+SecPerClust]
3551 0000C4AC 880D[3A8D0100]      <1>       mov   [DestinationFile_SecPerClust], cl
3552                              <1>
3553                              <1>       ; 17/03/2016
3554 0000C4B2 386F03              <1>       cmp   [edi+LD_FATType], ch ; 0
3555 0000C4B5 7707                <1>       ja    short csftdf2_set_df_percent_wsize1
3556                              <1>
3557 0000C4B7 B800000100          <1>       mov   eax, 65536 ; read/write buffer size for Singlix FS
3558 0000C4BC EB06                <1>       jmp   short csftdf2_set_df_percent_wsize2
3559                              <1>
3560                              <1> csftdf2_set_df_percent_wsize1:
3561 0000C4BE 0FB74711            <1>       movzx eax, word [edi+LD_BPB+BytesPerSec]
3562 0000C4C2 F7E1                <1>       mul   ecx
3563                              <1>       ;sub  edx, edx
3564                              <1> csftdf2_set_df_percent_wsize2:
3565 0000C4C4 A3[808D0100]        <1>       mov   [csftdf_w_size], eax
3566                              <1>
3567 0000C4C9 A1[688D0100]        <1>       mov   eax, [csftdf_filesize]
3568                              <1>
3569 0000C4CE 3D00000100          <1>       cmp   eax, 65536 ; 64KB   ; small file
3570 0000C4D3 721F                <1>       jb    short csftdf2_load_file ; do not display percentage
3571                              <1>
3572                              <1> csftdf2_reset_wf_percent_ptr_chk_64k:
3573 0000C4D5 B201                <1>       mov   dl, 1 ; 25/03/2016
3574                              <1>
3575 0000C4D7 3D00000400          <1>       cmp   eax, 65536*4 ; 256KB
3576 0000C4DC 7310                <1>       jnb   short csftdf2_enable_percentage_display ; big file
3577                              <1>
3578                              <1>       ; 64-128KB file size for floppy disks
3579 0000C4DE 3815[3C8C0100]      <1>       cmp   byte [SourceFile_Drv], dl ; 1 ; read from floppy disk ?
3580 0000C4E4 7608                <1>       jna   short csftdf2_enable_percentage_display
3581                              <1>
3582 0000C4E6 3815[BC8C0100]      <1>       cmp   byte [DestinationFile_Drv], dl ; 1 ; write to floppy disk ?
3583 0000C4EC 7706                <1>       ja    short csftdf2_load_file
3584                              <1>
3585                              <1> csftdf2_enable_percentage_display:
3586 0000C4EE 8815[8C8D0100]      <1>       mov   [csftdf_percentage], dl ; 1
3587                              <1>
3588                              <1> csftdf2_load_file:
3589                              <1>       ; 13/05/2016
3590                              <1>       ; 19/03/2016
3591                              <1>       ; 18/03/2016
3592                              <1>       ; 17/03/2016
3593 0000C4F4 B40F                <1>       mov   ah, 0Fh
3594 0000C4F6 E86E52FFFF          <1>       call  _int10h
3595                              <1>       ; 13/05/2016
3596 0000C4FB 883D[8D8D0100]      <1>       mov   [csftdf_videopage], bh ; active video page
3597 0000C501 B403                <1>       mov   ah, 03h
3598 0000C503 E86152FFFF          <1>       call  _int10h
3599 0000C508 668915[8E8D0100]    <1>       mov   [csftdf_cursorpos], dx
3600                              <1>
3601 0000C50F 29C0                <1>       sub   eax, eax
3602 0000C511 A2[658D0100]        <1>       mov   [csftdf_rw_err], al ; 0
3603                              <1>
3604                              <1> ; ///
3605                              <1> csftdf_sf_amb: ; 15/03/2016
```

```
3606 0000C516 8B0D[688D0100]      <1>       mov   ecx, [csftdf_filesize]    ; 23/03/2016
3607                              <1>
3608                              <1>       ; TRDOS 386 (TRDOS v2.0)
3609                              <1>       ; Allocate contiguous memory block for loading the file
3610                              <1>
3611                              <1>       ;mov   ecx, [SourceFile_DirEntry+DirEntry_FileSize]
3612                              <1>
3613                              <1>       ;sub   eax, eax ; First free memory aperture
3614                              <1>
3615                              <1>       ; eax = 0 (Allocate memory from the beginning)
3616                              <1>       ; ecx = File (Allocation) size in bytes
3617                              <1>
3618 0000C51C E8529FFFFF          <1>       call  allocate_memory_block
3619 0000C521 7304                <1>       jnc   short loc_check_sf_save_loading_parms
3620                              <1>
3621 0000C523 29C0                <1>       sub   eax, eax
3622 0000C525 29C9                <1>       sub   ecx, ecx
3623                              <1>
3624                              <1> loc_check_sf_save_loading_parms:
3625 0000C527 A3[6C8D0100]        <1>       mov   [csftdf_sf_mem_addr], eax ; loading address
3626 0000C52C 890D[708D0100]      <1>       mov   [csftdf_sf_mem_bsize], ecx ; block size
3627                              <1> ; ///
3628                              <1>       ; 19/03/2016
3629 0000C532 8B35[908D0100]      <1>       mov   esi, [csftdf_sf_drv_dt] ; logical dos drv desc. tbl.
3630                              <1>
3631                              <1>       ; 17/03/2016
3632 0000C538 09C0                <1>       or    eax, eax ; contiguous free memory block address
3633 0000C53A 0F845B010000        <1>         jz      csftdf2_read_sf_cluster
3634                              <1>
3635                              <1>       ; 18/03/2016
3636 0000C540 8B1D[6C8D0100]      <1>       mov   ebx, [csftdf_sf_mem_addr] ; memory block address
3637                              <1>
3638 0000C546 807E0300            <1>       cmp   byte [esi+LD_FATType], 0
3639 0000C54A 0F8605020000        <1>         jna     csftdf2_load_fs_file
3640                              <1>
3641                              <1> csftdf2_load_fat_file:
3642 0000C550 53                  <1>       push  ebx ; *
3643                              <1>
3644                              <1> csftdf2_load_fat_file_next:
3645 0000C551 BE[733E0100]        <1>       mov   esi, msg_reading
3646 0000C556 E8EAAFFFFF          <1>       call  print_msg
3647                              <1>
3648 0000C55B 803D[8C8D0100]00    <1>       cmp   byte [csftdf_percentage], 0
3649 0000C562 7605                <1>       jna   short csftdf2_load_fat_file_1
3650                              <1>
3651 0000C564 E87C000000          <1>       call  csftdf2_print_percentage ; 19/03/2016
3652                              <1>
3653                              <1> csftdf2_load_fat_file_1:
3654 0000C569 8B35[908D0100]      <1>       mov   esi, [csftdf_sf_drv_dt]
3655 0000C56F 5B                  <1>       pop   ebx ; *
3656                              <1>
3657                              <1> csftdf2_load_fat_file_2:
3658 0000C570 E8B8000000          <1>       call  csftdf2_read_fat_file_sectors ; 19/03/2016
3659 0000C575 0F8250040000        <1>         jc      csftdf2_rw_error ; eocc! or disk error!
3660                              <1>
3661 0000C57B 09D2                <1>       or    edx, edx ; edx > 0 -> EOF
3662 0000C57D 7520                <1>       jnz   short csftdf2_load_fat_file_ok
3663                              <1>
3664 0000C57F 803D[8C8D0100]00    <1>       cmp   byte [csftdf_percentage], 0
3665 0000C586 76E8                <1>       jna   short csftdf2_load_fat_file_2
3666                              <1>
3667 0000C588 53                  <1>       push  ebx ; *
3668                              <1>
3669                              <1>       ; Set cursor position
3670                              <1>       ; AH= 02h, BH= Page Number, DH= Row, DL= Column
3671 0000C589 8A3D[8D8D0100]      <1>       mov   bh, [csftdf_videopage]
3672 0000C58F 668B15[8E8D0100]    <1>       mov   dx, [csftdf_cursorpos]
3673 0000C596 B402                <1>       mov   ah, 2
3674 0000C598 E8CC51FFFF          <1>       call  _int10h
3675 0000C59D EBB2                <1>       jmp   short csftdf2_load_fat_file_next
3676                              <1>
3677                              <1> csftdf2_load_fat_file_ok:
3678 0000C59F 803D[8C8D0100]00    <1>       cmp   byte [csftdf_percentage], 0
3679 0000C5A6 0F8651020000        <1>         jna     csftdf2_save_file ; 25/03/2016
3680                              <1>
3681                              <1>       ; "Reading... 100%"
3682 0000C5AC BF[8B3E0100]        <1>       mov   edi, percentagestr
3683 0000C5B1 B031                <1>       mov   al, '1'
3684 0000C5B3 AA                  <1>       stosb
3685 0000C5B4 B030                <1>       mov   al, '0'
3686 0000C5B6 AA                  <1>       stosb
3687 0000C5B7 AA                  <1>       stosb
3688                              <1>
3689 0000C5B8 8A3D[8D8D0100]      <1>       mov   bh, [csftdf_videopage]
3690 0000C5BE 668B15[8E8D0100]    <1>       mov   dx, [csftdf_cursorpos]
3691 0000C5C5 B402                <1>       mov   ah, 2
3692 0000C5C7 E89D51FFFF          <1>       call  _int10h
3693                              <1>
3694 0000C5CC BE[733E0100]        <1>       mov   esi, msg_reading
3695 0000C5D1 E86FAFFFFF          <1>       call  print_msg
3696                              <1>
3697 0000C5D6 BE[8B3E0100]        <1>       mov   esi, percentagestr
3698 0000C5DB E865AFFFFF          <1>       call  print_msg
3699                              <1>
3700 0000C5E0 E918020000          <1>         jmp     csftdf2_save_file ; 25/03/2016
3701                              <1>
3702                              <1> csftdf2_print_percentage:
3703                              <1>       ; 09/12/2017
3704                              <1>       ; 19/03/2016
3705                              <1>       ; 18/03/2016
3706 0000C5E5 B020                <1>       mov   al, 20h
3707 0000C5E7 BF[8B3E0100]        <1>       mov   edi, percentagestr
3708 0000C5EC AA                  <1>       stosb
3709 0000C5ED AA                  <1>       stosb
3710 0000C5EE A1[848D0100]        <1>       mov   eax, [csftdf_sf_rbytes]
```

```
3711 0000C5F3 BA64000000          <1>       mov    edx, 100
3712 0000C5F8 F7E2                <1>       mul    edx
3713 0000C5FA 8B0D[688D0100]      <1>       mov    ecx, [csftdf_filesize]
3714 0000C600 F7F1                <1>       div    ecx
3715 0000C602 B10A                <1>       mov    cl, 10
3716 0000C604 F6F1                <1>       div    cl
3717 0000C606 80C430              <1>       add    ah, '0'
3718 0000C609 8827                <1>       mov    [edi], ah
3719 0000C60B 20C0                <1>       and    al, al
3720 0000C60D 740A                <1>       jz     short csftdf2_print_percent_1
3721 0000C60F 4F                  <1>       dec    edi
3722                              <1>       ;cbw
3723 0000C610 28E4                <1>       sub    ah, ah ; 09/12/2017
3724 0000C612 F6F1                <1>       div    cl
3725 0000C614 80C430              <1>       add    ah, '0'
3726 0000C617 8827                <1>       mov    [edi], ah
3727                              <1>       ;and   al, al
3728                              <1>       ;jz    short csftdf2_print_percent_1
3729                              <1>       ;dec   edi
3730                              <1>       ;mov   [edi], '1' ; 100%
3731                              <1>
3732                              <1> csftdf2_print_percent_1:
3733 0000C619 BE[8B3E0100]        <1>       mov    esi, percentagestr
3734                              <1>       ;call  print_msg
3735                              <1>       ;retn
3736 0000C61E E922AFFFFF          <1>       jmp    print_msg
3737                              <1>
3738                              <1> csftdf2_read_file_sectors:
3739                              <1>       ; 19/03/2016
3740 0000C623 807E0300            <1>       cmp    byte [esi+LD_FATType], 0
3741 0000C627 0F8627070000        <1>         jna     csftdf2_read_fs_file_sectors
3742                              <1>
3743                              <1> csftdf2_read_fat_file_sectors:
3744                              <1>       ; 19/03/2016
3745                              <1>       ; 18/03/2016
3746                              <1>       ; return:
3747                              <1>       ;   CF = 0 & EDX > 0 -> END OF FILE
3748                              <1>       ;   CF = 0 & EDX = 0 -> not EOF
3749                              <1>       ;   CF = 1 -> read error (error code in AL)
3750                              <1>
3751                              <1> csftdf2_read_fat_file_secs_0:
3752 0000C62D 8B15[688D0100]      <1>       mov    edx, [csftdf_filesize]
3753 0000C633 2B15[848D0100]      <1>       sub    edx, [csftdf_sf_rbytes]
3754 0000C639 3B15[7C8D0100]      <1>       cmp    edx, [csftdf_r_size]
3755 0000C63F 7306                <1>       jnb    short csftdf2_read_fat_file_secs_1
3756 0000C641 8915[7C8D0100]      <1>       mov    [csftdf_r_size], edx
3757                              <1>
3758                              <1> csftdf2_read_fat_file_secs_1:
3759 0000C647 A1[7C8D0100]        <1>       mov    eax, [csftdf_r_size]
3760 0000C64C 29D2                <1>       sub    edx, edx
3761 0000C64E 0FB74E11            <1>       movzx  ecx, word [esi+LD_BPB+BytesPerSec]
3762 0000C652 01C8                <1>       add    eax, ecx
3763 0000C654 48                  <1>       dec    eax
3764 0000C655 F7F1                <1>       div    ecx
3765 0000C657 89C1                <1>       mov    ecx, eax ; sector count
3766 0000C659 A1[748D0100]        <1>       mov    eax, [csftdf_sf_cluster]
3767                              <1>
3768                              <1>       ; EBX = memory block address (current)
3769                              <1>
3770 0000C65E E821090000          <1>       call   read_fat_file_sectors
3771 0000C663 7235                <1>       jc     short csftdf2_read_fat_file_secs_3
3772                              <1>
3773                              <1>       ; EBX = next memory address
3774                              <1>
3775 0000C665 A1[848D0100]        <1>       mov    eax, [csftdf_sf_rbytes]
3776 0000C66A 0305[7C8D0100]      <1>       add    eax, [csftdf_r_size]
3777 0000C670 8B15[688D0100]      <1>       mov    edx, [csftdf_filesize]
3778 0000C676 39D0                <1>       cmp    eax, edx
3779 0000C678 7320                <1>       jnb    short csftdf2_read_fat_file_secs_3 ; edx > 0
3780 0000C67A A3[848D0100]        <1>       mov    [csftdf_sf_rbytes], eax
3781                              <1>
3782 0000C67F 53                  <1>       push   ebx ; *
3783                              <1>       ; get next cluster (csftdf_r_size! bytes)
3784 0000C680 A1[748D0100]        <1>       mov    eax, [csftdf_sf_cluster]
3785 0000C685 E8CC060000          <1>       call   get_next_cluster
3786 0000C68A 5B                  <1>       pop    ebx ; *
3787 0000C68B 7306                <1>       jnc    short csftdf2_read_fat_file_secs_2
3788                              <1>
3789                              <1>       ; 15/10/2016
3790                              <1>       ;Disk read error instad of drv not ready err
3791 0000C68D B811000000          <1>       mov    eax, 17 ; Read error !
3792 0000C692 C3                  <1>       retn
3793                              <1>
3794                              <1> csftdf2_read_fat_file_secs_2:
3795 0000C693 29D2                <1>       sub    edx, edx ; 0
3796 0000C695 A3[748D0100]        <1>       mov    [csftdf_sf_cluster], eax ; next cluster
3797                              <1>
3798                              <1> csftdf2_read_fat_file_secs_3:
3799 0000C69A C3                  <1>       retn
3800                              <1>
3801                              <1> csftdf2_read_sf_cluster:
3802                              <1>       ; 19/03/2016
3803 0000C69B BB00000700          <1>       mov    ebx, Cluster_Buffer ; buffer address (64KB)
3804                              <1>
3805 0000C6A0 803D[8C8D0100]00    <1>       cmp    byte [csftdf_percentage], 0
3806 0000C6A7 760D                <1>       jna    short csftdf2_read_sf_clust_2
3807                              <1>
3808 0000C6A9 53                  <1>       push   ebx ; *
3809                              <1>
3810                              <1> csftdf2_read_sf_clust_next:
3811 0000C6AA E836FFFFFF          <1>       call   csftdf2_print_percentage
3812                              <1>
3813                              <1> csftdf2_read_sf_clust_0:
3814 0000C6AF 8B35[908D0100]      <1>       mov    esi, [csftdf_sf_drv_dt]
3815                              <1> csftdf2_read_sf_clust_1:
```

```
3816 0000C6B5 5B                <1>       pop    ebx ; *
3817                            <1>
3818                            <1> csftdf2_read_sf_clust_2:
3819 0000C6B6 89DA              <1>       mov    edx, ebx
3820 0000C6B8 0315[7C8D0100]    <1>       add    edx, [csftdf_r_size]
3821 0000C6BE 81FA00000800      <1>       cmp    edx, Cluster_Buffer + 65536
3822 0000C6C4 772F              <1>       ja     short csftdf2_write_df_cluster
3823                            <1>
3824 0000C6C6 E858FFFFFF        <1>       call   csftdf2_read_file_sectors ; 19/03/2016
3825 0000C6CB 0F8280020000      <1>         jc     csftdf2_save_fat_file_err2 ; eocc! or disk error!
3826                            <1>
3827 0000C6D1 09D2              <1>       or     edx, edx ; edx > 0 -> EOF
3828 0000C6D3 7520              <1>       jnz    short csftdf2_write_df_cluster
3829                            <1>
3830 0000C6D5 803D[8C8D0100]00  <1>       cmp    byte [csftdf_percentage], 0
3831 0000C6DC 76D8              <1>       jna    short csftdf2_read_sf_clust_2
3832                            <1>
3833 0000C6DE 53                <1>       push   ebx ; *
3834                            <1>
3835                            <1>       ; Set cursor position
3836                            <1>       ; AH= 02h, BH= Page Number, DH= Row, DL= Column
3837 0000C6DF 8A3D[8D8D0100]    <1>       mov    bh, [csftdf_videopage]
3838 0000C6E5 668B15[8E8D0100]  <1>       mov    dx, [csftdf_cursorpos]
3839 0000C6EC B402              <1>       mov    ah, 2
3840 0000C6EE E87650FFFF        <1>       call   _int10h
3841 0000C6F3 EBB5              <1>       jmp    short csftdf2_read_sf_clust_next
3842                            <1>
3843                            <1> csftdf2_write_df_cluster:
3844                            <1>       ; 19/03/2016
3845 0000C6F5 8B35[948D0100]    <1>       mov    esi, [csftdf_df_drv_dt]
3846 0000C6FB BB00000700        <1>       mov    ebx, Cluster_Buffer ; buffer address (64KB)
3847                            <1>
3848                            <1> csftdf2_write_df_clust_next:
3849 0000C700 E855000000        <1>       call   csftdf2_write_file_sectors ; 19/03/2016
3850 0000C705 0F8246020000      <1>         jc     csftdf2_save_fat_file_err2 ; eocc! or disk error!
3851                            <1>
3852 0000C70B 09D2              <1>       or     edx, edx ; edx > 0 -> EOF
3853 0000C70D 750A              <1>       jnz    short csftdf2_rw_f_clust_ok
3854                            <1>
3855 0000C70F 81FB00000800      <1>       cmp    ebx, Cluster_Buffer + 65536
3856 0000C715 72E9              <1>       jb     short csftdf2_write_df_clust_next
3857                            <1>
3858 0000C717 EB82              <1>       jmp    short csftdf2_read_sf_cluster
3859                            <1>
3860                            <1> csftdf2_rw_f_clust_ok:
3861 0000C719 803D[8C8D0100]00  <1>       cmp    byte [csftdf_percentage], 0
3862 0000C720 0F86B2010000      <1>         jna    csftdf2_save_fat_file_4 ; 25/03/2016
3863                            <1>
3864                            <1>       ; "100%"
3865 0000C726 BF[8B3E0100]      <1>       mov    edi, percentagestr
3866 0000C72B B031              <1>       mov    al, '1'
3867 0000C72D AA                <1>       stosb
3868 0000C72E B030              <1>       mov    al, '0'
3869 0000C730 AA                <1>       stosb
3870 0000C731 AA                <1>       stosb
3871                            <1>
3872 0000C732 8A3D[8D8D0100]    <1>       mov    bh, [csftdf_videopage]
3873 0000C738 668B15[8E8D0100]  <1>       mov    dx, [csftdf_cursorpos]
3874 0000C73F B402              <1>       mov    ah, 2
3875 0000C741 E82350FFFF        <1>       call   _int10h
3876                            <1>
3877 0000C746 BE[8B3E0100]      <1>       mov    esi, percentagestr
3878 0000C74B E8F5ADFFFF        <1>       call   print_msg
3879                            <1>
3880 0000C750 E983010000        <1>         jmp    csftdf2_save_fat_file_4
3881                            <1>
3882                            <1> csftdf2_load_fs_file:
3883                            <1>       ; temporary - 18/03/2016
3884 0000C755 E96F020000        <1>         jmp    csftdf2_read_error
3885                            <1>
3886                            <1> csftdf2_write_file_sectors:
3887                            <1>       ; 19/03/2016
3888 0000C75A 807E0300          <1>       cmp    byte [esi+LD_FATType], 0
3889 0000C75E 0F86F1050000      <1>         jna    csftdf2_write_fs_file_sectors
3890                            <1>
3891                            <1> csftdf2_write_fat_file_sectors:
3892                            <1>       ; 19/03/2016
3893                            <1>       ; 18/03/2016
3894                            <1>       ; return:
3895                            <1>       ;   CF = 0 & EDX > 0 -> END OF FILE
3896                            <1>       ;   CF = 0 & EDX = 0 -> not EOF
3897                            <1>       ;   CF = 1 -> write error (error code in AL)
3898                            <1>
3899                            <1> csftdf2_write_fat_file_secs_0:
3900 0000C764 8B15[688D0100]      <1>       mov    edx, [csftdf_filesize]
3901 0000C76A 2B15[888D0100]      <1>       sub    edx, [csftdf_df_wbytes]
3902 0000C770 3B15[808D0100]      <1>       cmp    edx, [csftdf_w_size]
3903 0000C776 7306                <1>       jnb    short csftdf2_write_fat_file_secs_1
3904 0000C778 8915[808D0100]      <1>       mov    [csftdf_w_size], edx
3905                              <1>
3906                              <1> csftdf2_write_fat_file_secs_1:
3907 0000C77E A1[808D0100]        <1>       mov    eax, [csftdf_w_size]
3908 0000C783 29D2                <1>       sub    edx, edx
3909 0000C785 0FB74E11            <1>       movzx  ecx, word [esi+LD_BPB+BytesPerSec]
3910 0000C789 01C8                <1>       add    eax, ecx
3911 0000C78B 48                  <1>       dec    eax
3912 0000C78C F7F1                <1>       div    ecx
3913 0000C78E 89C1                <1>       mov    ecx, eax ; sector count
3914 0000C790 A1[788D0100]        <1>       mov    eax, [csftdf_df_cluster]
3915                              <1>
3916                              <1>       ; EBX = memory block address (current)
3917                              <1>
3918 0000C795 E8A20F0000          <1>       call   write_fat_file_sectors
3919 0000C79A 7259                <1>       jc     short csftdf2_write_fat_file_secs_4
3920                              <1>
```

```
                                <1>       ; EBX = next memory address
                                <1>
0000C79C A1[888D0100]           <1>       mov    eax, [csftdf_df_wbytes]
0000C7A1 0305[808D0100]         <1>       add    eax, [csftdf_w_size]
0000C7A7 8B15[688D0100]         <1>       mov    edx, [csftdf_filesize]
0000C7AD 39D0                   <1>       cmp    eax, edx
0000C7AF 7344                   <1>       jnb    short csftdf2_write_fat_file_secs_4
0000C7B1 A3[888D0100]           <1>       mov    [csftdf_df_wbytes], eax
                                <1>       ;
0000C7B6 A3[2A8D0100]           <1>       mov    [DestinationFile_DirEntry+DirEntry_FileSize], eax
                                <1>
0000C7BB 53                     <1>       push   ebx ; *
                                <1>
0000C7BC 803D[668D0100]01       <1>       cmp    byte [DestinationFileFound], 1
0000C7C3 7210                   <1>       jb     short csftdf2_write_fat_file_secs_2
                                <1>
                                <1>       ; get next cluster (csftdf_w_size! bytes)
0000C7C5 A1[788D0100]           <1>       mov    eax, [csftdf_df_cluster]
0000C7CA E887050000             <1>       call   get_next_cluster
0000C7CF 731C                   <1>       jnc    short csftdf2_write_fat_file_secs_3
                                <1>
0000C7D1 21C0                   <1>       and    eax, eax ; end of cluster chain!?
0000C7D3 7521                   <1>       jnz    short csftdf2_write_fat_file_secs_5 ; disk error !
                                <1>
                                <1> csftdf2_write_fat_file_secs_2:
0000C7D5 A1[788D0100]           <1>       mov    eax, [csftdf_df_cluster] ; last cluster
0000C7DA E8800E0000             <1>       call   add_new_cluster
0000C7DF 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!)
0000C7E1 8B15[808D0100]         <1>       mov    edx, [csftdf_w_size] ; bytes per cluster
0000C7E7 0115[2A8D0100]         <1>       add    [DestinationFile_DirEntry+DirEntry_FileSize], edx
                                <1>
                                <1> csftdf2_write_fat_file_secs_3:
0000C7ED 5B                     <1>       pop    ebx ; *
0000C7EE 29D2                   <1>       sub    edx, edx ; 0
0000C7F0 A3[788D0100]           <1>       mov    [csftdf_df_cluster], eax ; next cluster
                                <1>
                                <1> csftdf2_write_fat_file_secs_4:
0000C7F5 C3                     <1>       retn
                                <1>
                                <1> csftdf2_write_fat_file_secs_5:
0000C7F6 5B                     <1>       pop    ebx ; *
                                <1>       ; 16/10/2016 (1Dh -> 18)
0000C7F7 B812000000             <1>       mov    eax, 18 ; Write error !
0000C7FC C3                     <1>       retn
                                <1>
                                <1> csftdf2_save_file:
                                <1>       ; 09/12/2017
                                <1>       ; 25/03/2016
                                <1>       ; 19/03/2016
                                <1>       ; 18/03/2016
0000C7FD 8B35[948D0100]         <1>       mov    esi, [csftdf_df_drv_dt] ; logical dos drv desc. tbl.
                                <1>
0000C803 8B1D[6C8D0100]         <1>       mov    ebx, [csftdf_sf_mem_addr] ; memory block address
                                <1>
0000C809 807E0300               <1>       cmp    byte [esi+LD_FATType], 0
0000C80D 0F86F4010000           <1>         jna     csftdf2_save_fs_file
                                <1>
                                <1> csftdf2_save_fat_file:
0000C813 53                     <1>       push   ebx; *
                                <1>
0000C814 803D[8C8D0100]00       <1>       cmp    byte [csftdf_percentage], 0
0000C81B 7724                   <1>       ja     short csftdf2_save_fat_file_0
                                <1>
                                <1>       ; Set cursor position
                                <1>       ; AH= 02h, BH= Page Number, DH= Row, DL= Column
0000C81D 8A3D[8D8D0100]         <1>       mov    bh, [csftdf_videopage]
0000C823 668B15[8E8D0100]       <1>       mov    dx, [csftdf_cursorpos]
0000C82A B402                   <1>       mov    ah, 2
0000C82C E8384FFFFF             <1>       call   _int10h
                                <1>
0000C831 BE[7F3E0100]           <1>       mov    esi, msg_writing
0000C836 E80AADFFFF             <1>       call   print_msg
                                <1>
                                <1> csftdf2_save_fat_file_next:
0000C83B 8B35[948D0100]         <1>       mov    esi, [csftdf_df_drv_dt] ; 25/03/2016
                                <1>
                                <1> csftdf2_save_fat_file_0:
0000C841 5B                     <1>       pop    ebx ; *
                                <1>
                                <1> csftdf2_save_fat_file_1:
0000C842 E813FFFFFF             <1>       call   csftdf2_write_file_sectors ; 19/03/2016
0000C847 0F827E010000           <1>         jc      csftdf2_rw_error ; eocc! or disk error!
                                <1>
0000C84D 09D2                   <1>       or     edx, edx ; edx > 0 -> EOF
0000C84F 756D                   <1>         jnz     short csftdf2_save_fat_file_3 ; 25/03/2016
                                <1>
0000C851 803D[8C8D0100]00       <1>       cmp    byte [csftdf_percentage], 0
0000C858 76E8                   <1>       jna    short csftdf2_save_fat_file_1
                                <1>
0000C85A B020                   <1>       mov    al, 20h
0000C85C BF[8B3E0100]           <1>       mov    edi, percentagestr
0000C861 AA                     <1>       stosb
0000C862 AA                     <1>       stosb
0000C863 A1[888D0100]           <1>       mov    eax, [csftdf_df_wbytes]
0000C868 BA64000000             <1>       mov    edx, 100
0000C86D F7E2                   <1>       mul    edx
0000C86F 8B0D[688D0100]         <1>       mov    ecx, [csftdf_filesize]
0000C875 F7F1                   <1>       div    ecx
0000C877 B10A                   <1>       mov    cl, 10
```

```
4026 0000C879 F6F1              <1>       div    cl
4027 0000C87B 80C430            <1>       add    ah, '0'
4028 0000C87E 8827              <1>       mov    [edi], ah
4029 0000C880 20C0              <1>       and    al, al
4030 0000C882 740A              <1>       jz     short csftdf2_save_fat_file_2
4031 0000C884 4F                <1>       dec    edi
4032                            <1>       ;cbw
4033 0000C885 30E4              <1>       xor    ah, ah ; 09/12/2017
4034 0000C887 F6F1              <1>       div    cl
4035 0000C889 80C430            <1>       add    ah, '0'
4036 0000C88C 8827              <1>       mov    [edi], ah
4037                            <1>       ;and   al, al
4038                            <1>       ;jz    short csftdf2_save_fat_file_2
4039                            <1>       ;dec   edi
4040                            <1>       ;mov   [edi], '1' ; 100%
4041                            <1>
4042                            <1> csftdf2_save_fat_file_2:
4043 0000C88E 53                <1>       push   ebx ; *
4044                            <1>
4045 0000C88F E802000000        <1>       call   csftdf2_print_wr_percentage ; 25/03/2016
4046                            <1>
4047 0000C894 EBA5              <1>         jmp     csftdf2_save_fat_file_next
4048                            <1>
4049                            <1> csftdf2_print_wr_percentage:
4050                            <1>       ; Set cursor position
4051                            <1>       ; AH= 02h, BH= Page Number, DH= Row, DL= Column
4052 0000C896 8A3D[8D8D0100]    <1>       mov    bh, [csftdf_videopage]
4053 0000C89C 668B15[8E8D0100]  <1>       mov    dx, [csftdf_cursorpos]
4054 0000C8A3 B402              <1>       mov    ah, 2
4055 0000C8A5 E8BF4EFFFF        <1>       call   _int10h
4056                            <1>
4057 0000C8AA BE[7F3E0100]      <1>       mov    esi, msg_writing
4058 0000C8AF E891ACFFFF        <1>       call   print_msg
4059                            <1>
4060 0000C8B4 BE[8B3E0100]      <1>       mov    esi, percentagestr
4061                            <1>       ;call  print_msg
4062                            <1>       ;retn
4063 0000C8B9 E987ACFFFF        <1>       jmp    print_msg
4064                            <1>
4065                            <1> csftdf2_save_fat_file_3:
4066 0000C8BE 803D[8C8D0100]00  <1>       cmp    byte [csftdf_percentage], 0
4067 0000C8C5 7611              <1>         jna     csftdf2_save_fat_file_4 ; 25/03/2016
4068                            <1>
4069                            <1>       ; "100%"
4070 0000C8C7 BF[8B3E0100]      <1>       mov    edi, percentagestr
4071 0000C8CC B031              <1>       mov    al, '1'
4072 0000C8CE AA                <1>       stosb
4073 0000C8CF B030              <1>       mov    al, '0'
4074 0000C8D1 AA                <1>       stosb
4075 0000C8D2 AA                <1>       stosb
4076                            <1>
4077 0000C8D3 E8BEFFFFFF        <1>       call   csftdf2_print_wr_percentage
4078                            <1>
4079                            <1> csftdf2_save_fat_file_4:
4080 0000C8D8 803D[668D0100]00  <1>       cmp    byte [DestinationFileFound], 0
4081 0000C8DF 7647              <1>       jna    short csftdf2_save_fat_file_6
4082                            <1>
4083 0000C8E1 8B35[948D0100]    <1>       mov    esi, [csftdf_df_drv_dt] ; 31/03/2016
4084                            <1>
4085 0000C8E7 A1[788D0100]      <1>       mov    eax, [csftdf_df_cluster] ; last cluster
4086 0000C8EC E865040000        <1>       call   get_next_cluster
4087 0000C8F1 7235              <1>       jc     short csftdf2_save_fat_file_6 ; eocc! or disk error!
4088                            <1>
4089 0000C8F3 A1[788D0100]      <1>       mov    eax, [csftdf_df_cluster] ; last cluster
4090                            <1>       ;xor   ecx, ecx
4091                            <1>       ;mov   [FAT_ClusterCounter], ecx ; 0 ; reset
4092                            <1>       ;dec   ecx ; 0FFFFFFFFh
4093                            <1>       ;shr   ecx, 4 ; 28 bit ; 0FFFFFFFh
4094 0000C8F8 B9FFFFFF0F        <1>       mov    ecx, 0FFFFFFFh
4095 0000C8FD E87E070000        <1>       call   update_cluster
4096 0000C902 7224              <1>       jc     short csftdf2_save_fat_file_6 ; really last cluster!?
4097                            <1>
4098 0000C904 A3[788D0100]      <1>       mov    [csftdf_df_cluster], eax ; next cluster
4099                            <1>
4100                            <1>       ; byte [FAT_BuffValidData] = 2
4101 0000C909 E82F0A0000        <1>       call   save_fat_buffer
4102 0000C90E 730E              <1>       jnc    short csftdf2_save_fat_file_5
4103                            <1>
4104 0000C910 8B15[688D0100]    <1>       mov    edx, [csftdf_filesize]
4105 0000C916 8915[2A8D0100]    <1>       mov    [DestinationFile_DirEntry+DirEntry_FileSize], edx
4106 0000C91C EB58              <1>       jmp    short csftdf2_save_fat_file_err3
4107                            <1>
4108                            <1> csftdf2_save_fat_file_5:
4109 0000C91E A1[788D0100]      <1>       mov    eax, [csftdf_df_cluster]
4110                            <1>
4111                            <1>       ; EAX = First cluster to be truncated/unlinked
4112                            <1>       ; ESI = Logical dos drive description table address
4113 0000C923 E8580C0000        <1>       call   truncate_cluster_chain
4114                            <1>
4115                            <1> csftdf2_save_fat_file_6:
4116                            <1>       ; 28/03/2016
4117 0000C928 BE[998C0100]      <1>       mov    esi, SourceFile_DirEntry+DirEntry_Attr ; +11 to + 18
4118 0000C92D BF[198D0100]      <1>       mov    edi, DestinationFile_DirEntry+DirEntry_Attr ; +11 to + 18
4119 0000C932 A4                <1>       movsb ; +11
4120 0000C933 A5                <1>       movsd ; +12 .. +15
4121 0000C934 66A5              <1>       movsw ; +16 .. +17
4122                            <1>             ; + 18
4123 0000C936 83C604            <1>       add    esi, 4
4124 0000C939 83C704            <1>       add    edi, 4
4125 0000C93C A5                <1>       movsd  ; DirEntry_WrtTime ; +22 .. +25
4126                            <1>
4127 0000C93D 8B15[688D0100]    <1>       mov    edx, [csftdf_filesize]
4128 0000C943 8915[2A8D0100]    <1>       mov    [DestinationFile_DirEntry+DirEntry_FileSize], edx
4129                            <1>
4130 0000C949 E8BAF0FFFF        <1>       call   convert_current_date_time
```

```
                                <1>       ; DX = Date in dos dir entry format
                                <1>       ; AX = Time in dos dir entry format
0000C94E EB4D                   <1>       jmp    short csftdf2_save_fat_file_7
                                <1>
                                <1> csftdf2_save_fat_file_err1:
0000C950 5B                     <1>       pop    ebx ; *
                                <1> csftdf2_save_fat_file_err2:
0000C951 A1[888D0100]           <1>       mov    eax, [csftdf_df_wbytes]
0000C956 8B15[2A8D0100]         <1>       mov    edx, [DestinationFile_DirEntry+DirEntry_FileSize]
0000C95C 39C2                   <1>       cmp    edx, eax
0000C95E 7616                   <1>       jna    short csftdf2_save_fat_file_err3
0000C960 A1[788D0100]           <1>       mov    eax, [csftdf_df_cluster] ; last (empty) cluster
                                <1>       ; ESI = Logical dos drive description table address
0000C965 E8160C0000             <1>       call   truncate_cluster_chain
0000C96A 720A                   <1>       jc     short csftdf2_save_fat_file_err3
0000C96C A1[888D0100]           <1>       mov    eax, [csftdf_df_wbytes]
0000C971 A3[2A8D0100]           <1>       mov    [DestinationFile_DirEntry+DirEntry_FileSize], eax
                                <1> csftdf2_save_fat_file_err3:
0000C976 E88DF0FFFF             <1>       call   convert_current_date_time
                                <1>       ; DX = Date in dos dir entry format
                                <1>       ; AX = Time in dos dir entry format
0000C97B C605[1B8D0100]00       <1>       mov    byte [DestinationFile_DirEntry+DirEntry_CrtTimeTenth], 0
0000C982 66A3[1C8D0100]         <1>       mov    [DestinationFile_DirEntry+DirEntry_CrtTime], ax
0000C988 668915[1E8D0100]       <1>       mov    [DestinationFile_DirEntry+DirEntry_CrtDate], dx
0000C98F 66A3[248D0100]         <1>       mov    [DestinationFile_DirEntry+DirEntry_WrtTime], ax
0000C995 668915[268D0100]       <1>       mov    [DestinationFile_DirEntry+DirEntry_WrtDate], dx
0000C99C F9                     <1>       stc
                                <1> csftdf2_save_fat_file_7:
0000C99D 9C                     <1>       pushf
0000C99E 668915[208D0100]       <1>       mov    [DestinationFile_DirEntry+DirEntry_LastAccDate], dx
0000C9A5 BE[0E8D0100]           <1>       mov    esi, DestinationFile_DirEntry
0000C9AA BF00000800             <1>       mov    edi, Directory_Buffer
0000C9AF 0FB70D[368D0100]       <1>       movzx  ecx, word [DestinationFile_DirEntryNumber] ; (<2048)
0000C9B6 66C1E105               <1>       shl    cx, 5 ; 32 * directory entry number
0000C9BA 01CF                   <1>       add    edi, ecx
                                <1>       ;mov   ecx, 8
0000C9BC 66B90800               <1>       mov    cx, 8
0000C9C0 F3A5                   <1>       rep    movsd
0000C9C2 9D                     <1>       popf
0000C9C3 730B                   <1>       jnc    short csftdf2_write_file_OK
                                <1>
                                <1> csftdf2_write_error:
                                <1>       ; 18/03/2016
0000C9C5 B01D                   <1>       mov    al, 1Dh ; write error
0000C9C7 EB02                   <1>       jmp    short csftdf2_rw_error
                                <1>
                                <1>       ; 16/03/2016
                                <1> csftdf2_read_error:
0000C9C9 B011                   <1>       mov    al, 17 ; ; Drive not ready or read error!
                                <1> csftdf2_rw_error:
0000C9CB A2[658D0100]           <1>       mov    [csftdf_rw_err], al
                                <1>
                                <1> csftdf2_write_file_OK:
                                <1>       ; 18/03/2016
0000C9D0 C605[7C890100]02       <1>       mov    byte [DirBuff_ValidData], 2
0000C9D7 E8CAF0FFFF             <1>       call   save_directory_buffer
                                <1>
                                <1>       ; Update last modification date&time of destination
                                <1>       ; file's (parent) directory
0000C9DC E860F1FFFF             <1>       call   update_parent_dir_lmdt
                                <1>       ;
0000C9E1 A1[6C8D0100]           <1>       mov    eax, [csftdf_sf_mem_addr] ; start address
                                <1>
0000C9E6 21C0                   <1>       and    eax, eax
0000C9E8 750E                   <1>       jnz    short csftdf2_dealloc_mblock
                                <1>
0000C9EA 88C5                   <1>       mov    ch, al ; 0 (Cluster r/w, not full loading)
                                <1> csftdf2_dealloc_retn:
0000C9EC 8A0D[658D0100]         <1>       mov    cl, [csftdf_rw_err]
0000C9F2 A1[788D0100]           <1>       mov    eax, [csftdf_df_cluster]
0000C9F7 C3                     <1>       retn
                                <1>
                                <1> csftdf2_dealloc_mblock:
0000C9F8 8B0D[708D0100]         <1>       mov    ecx, [csftdf_sf_mem_bsize] ; block size
0000C9FE E87D9CFFFF             <1>       call   deallocate_memory_block
0000CA03 B5FF                   <1>         mov     ch, 0FFh ; (File was full loaded at memory)
0000CA05 EBE5                   <1>       jmp    short csftdf2_dealloc_retn
                                <1>
                                <1> csftdf2_save_fs_file:
                                <1>       ; 16/10/2016 (1Dh -> 18)
                                <1>       ; temporary - (21/03/2016)
0000CA07 B812000000             <1>       mov    eax, 18 ; write error
0000CA0C F9                     <1>       stc
0000CA0D C3                     <1>       retn
                                <1>
                                <1> create_file:
                                <1>       ; 16/10/2016
                                <1>       ; 24/03/2016, 31/03/2016
                                <1>       ; 20/03/2016, 21/03/2016, 23/03/2016
                                <1>       ; 19/03/2016 (TRDOS 396 = TRDOS v2.0)
                                <1>       ; 03/09/2011 (FILE.ASM, 'proc_create_file')
                                <1>       ; 09/08/2010
                                <1>       ;
                                <1>       ; INPUT ->
                                <1>       ;     EAX = File Size
                                <1>       ;     ESI = ASCIIZ File Name
                                <1>       ;     CL = File Attributes
                                <1>       ;     EBX = FFFFFFFFh -> create empty file
                                <1>       ;                  (only for FAT fs)
                                <1>       ; OUTPUT ->
                                <1>       ;    CF = 0 ->
                                <1>       ;     EAX = New file's first cluster
                                <1>       ;     ESI = Logical Dos Drv Descr. Table Addr.
                                <1>       ;     EBX = offset CreateFile_Size
                                <1>       ;     ECX = Sectors per cluster (<256)
```

```
4236                                 <1>       ;      EDX = Directory entry index/number (<65536)
4237                                 <1>       ;     CF = 1 -> error code in AL
4238                                 <1>
4239                                 <1> ;     test   cl, 18h (directory or volume name)
4240                                 <1> ;     jnz    short loc_createfile_access_denied
4241 0000CA0E 80E107                <1>       and    cl, 07h ; S, H, R
4242 0000CA11 880D[B48D0100]        <1>         mov     [createfile_attrib], cl
4243                                 <1>
4244 0000CA17 89D9                  <1>       mov    ecx, ebx
4245 0000CA19 89F3                  <1>       mov    ebx, esi ; ASCIIZ File Name address
4246 0000CA1B 29D2                  <1>       sub    edx, edx
4247 0000CA1D 8A35[56820100]        <1>         mov     dh, [Current_Drv]
4248 0000CA23 BE00010900            <1>         mov     esi, Logical_DOSDisks
4249 0000CA28 01D6                  <1>       add    esi, edx
4250                                 <1>
4251 0000CA2A 8815[BF8D0100]        <1>       mov    [createfile_UpdatePDir], dl ; 0 ; 31/03/2016
4252                                 <1>
4253                                 <1>       ; LD_DiskType = 0 for write protection (read only)
4254 0000CA30 807E0101              <1>       cmp    byte [esi+LD_DiskType], 1 ; 0 = Invalid
4255 0000CA34 730A                  <1>       jnb    short loc_createfile_check_file_sytem
4256                                 <1>       ; 16/10/2016 (TRDOS Error code: 30, disk write protected)
4257 0000CA36 B81E000000            <1>       mov    eax, 30 ; 13h, MSDOS err : Disk write-protected
4258 0000CA3B 66BA0000              <1>       mov    dx, 0
4259                                 <1>       ; err retn: EDX = 0, EBX = File name offset
4260                                 <1>       ; ESI -> Dos drive description table address
4261 0000CA3F C3                    <1>       retn
4262                                 <1>
4263                                 <1> ;loc_createfile_access_denied:
4264                                 <1> ;     mov    eax, 05h ; access denied (invalid attributes input)
4265                                 <1> ;     stc
4266                                 <1> ;     retn
4267                                 <1>
4268                                 <1> loc_createfile_check_file_sytem:
4269 0000CA40 807E0301              <1>       cmp    byte [esi+LD_FATType], 1
4270 0000CA44 730A                  <1>       jnb    short loc_createfile_chk_empty_FAT_file_sign1
4271                                 <1>
4272 0000CA46 A3[A08D0100]          <1>       mov    [createfile_size], eax
4273                                 <1>       ; ESI = Logical Dos Drive Description Table address
4274                                 <1>       ; EBX = ASCIIZ File Name address
4275 0000CA4B E9FE020000            <1>       jmp    create_fs_file
4276                                 <1>
4277                                 <1> loc_createfile_chk_empty_FAT_file_sign1:
4278                                 <1>       ; ECX = FFFFFFFFh -> create empty file if drive has FAT fs
4279 0000CA50 41                    <1>       inc    ecx
4280 0000CA51 7506                  <1>       jnz    short loc_createfile_chk_empty_FAT_file_sign2
4281 0000CA53 890D[A08D0100]        <1>       mov    [createfile_size], ecx ; 0 ; empty file
4282                                 <1>
4283                                 <1> loc_createfile_chk_empty_FAT_file_sign2:
4284                                 <1>       ; 23/03/2016
4285 0000CA59 668B4E11              <1>       mov    cx, [esi+LD_BPB+BytesPerSec]
4286 0000CA5D 66890D[BC8D0100]      <1>       mov    [createfile_BytesPerSec], cx
4287                                 <1>
4288                                 <1>       ; EBX = ASCIIZ File Name address
4289 0000CA64 0FB65613              <1>       movzx  edx, byte [esi+LD_BPB+SecPerClust]
4290 0000CA68 8815[B58D0100]        <1>       mov    [createfile_SecPerClust], dl
4291 0000CA6E 8B4E74                <1>       mov    ecx, [esi+LD_FreeSectors]
4292 0000CA71 39D1                  <1>       cmp    ecx, edx ; byte [createfile_SecPerClust]
4293 0000CA73 7306                  <1>       jnb    short loc_create_fat_file
4294                                 <1>
4295                                 <1> loc_createfile_insufficient_disk_space:
4296 0000CA75 B827000000            <1>       mov    eax, 27h
4297                                 <1> loc_createfile_gffc_retn:
4298 0000CA7A C3                    <1>       retn
4299                                 <1>
4300                                 <1> loc_create_fat_file:
4301 0000CA7B 891D[988D0100]        <1>       mov    [createfile_Name_Offset], ebx
4302 0000CA81 890D[9C8D0100]        <1>       mov    [createfile_FreeSectors], ecx
4303                                 <1>
4304                                 <1> loc_createfile_gffc_1:
4305 0000CA87 E821050000            <1>       call   get_first_free_cluster
4306 0000CA8C 72EC                  <1>       jc     short loc_createfile_gffc_retn
4307                                 <1>
4308 0000CA8E A3[A48D0100]          <1>       mov    [createfile_FFCluster], eax
4309                                 <1>
4310                                 <1> loc_createfile_locate_ffe_on_directory:
4311                                 <1>       ; Current directory fcluster <> Directory buffer cluster
4312                                 <1>       ; Current directory will be reloaded by
4313                                 <1>       ; 'locate_current_dir_file' procedure
4314                                 <1>       ;
4315                                 <1>       ; ESI = Logical Dos Drv Desc. Table Adress
4316 0000CA93 56                    <1>       push   esi ; *
4317 0000CA94 31C0                  <1>       xor    eax, eax
4318                                 <1>
4319 0000CA96 A3[72890100]          <1>       mov    dword [FAT_ClusterCounter], eax ; 0
4320                                 <1>       ; 21/03/2016
4321 0000CA9B A2[BE8D0100]          <1>       mov    byte [createfile_wfc], al ; 0
4322                                 <1>
4323 0000CAA0 89C1                  <1>       mov    ecx, eax
4324 0000CAA2 6649                  <1>       dec    cx ; FFFFh
4325                                 <1>       ; CX = FFFFh -> find first deleted or free entry
4326                                 <1>       ; ESI would be ASCIIZ filename address if the call
4327                                 <1>       ; would not be for first free or deleted dir entry
4328 0000CAA4 E8D7E7FFFF            <1>       call   locate_current_dir_file
4329 0000CAA9 0F83EE000000          <1>       jnc    loc_createfile_set_ff_dir_entry
4330 0000CAAF 5E                    <1>       pop    esi ; *
4331                                 <1>        ; ESI = Logical DOS Drv. Description Table Address
4332 0000CAB0 83F802                <1>       cmp    eax, 2
4333 0000CAB3 7402                  <1>       je     short loc_createfile_add_new_cluster
4334                                 <1> loc_createfile_locate_file_stc_retn:
4335 0000CAB5 F9                    <1>       stc
4336 0000CAB6 C3                    <1>       retn
4337                                 <1>
4338                                 <1> loc_createfile_add_new_cluster:
4339 0000CAB7 803D[55820100]02      <1>       cmp    byte [Current_FATType], 2
4340                                 <1>       ;cmp   byte [esi+LD_FATType], 2
```

```
4341 0000CABE 770C                <1>       ja     short loc_createfile_add_new_cluster_check_fsc
4342 0000CAC0 803D[54820100]01    <1>       cmp    byte [Current_Dir_Level], 1
4343                              <1>       ;cmp   byte [esi+LD_CDirLevel], 1
4344 0000CAC7 7303                <1>       jnb    short loc_createfile_add_new_cluster_check_fsc
4345                              <1>
4346                              <1>       ;mov   eax, 12
4347 0000CAC9 B00C                <1>       mov    al, 12 ; No more files
4348                              <1>
4349                              <1> loc_createfile_anc_retn:
4350 0000CACB C3                  <1>       retn
4351                              <1>
4352                              <1> loc_createfile_add_new_cluster_check_fsc:
4353 0000CACC 8B0D[9C8D0100]      <1>       mov    ecx, [createfile_FreeSectors]
4354 0000CAD2 0FB605[B58D0100]    <1>       movzx  eax, byte [createfile_SecPerClust]
4355 0000CAD9 66D1E0              <1>       shl    ax, 1 ; AX = 2 * AX
4356 0000CADC 39C1                <1>       cmp    ecx, eax
4357 0000CADE 7295                <1>         jb   short loc_createfile_insufficient_disk_space
4358                              <1>
4359                              <1> loc_createfile_add_new_subdir_cluster:
4360 0000CAE0 8B15[81890100]      <1>       mov    edx, [DirBuff_Cluster]
4361 0000CAE6 8915[A88D0100]      <1>       mov    [createfile_LastDirCluster], edx
4362                              <1>
4363 0000CAEC A1[A48D0100]        <1>       mov    eax, [createfile_FFCluster]
4364 0000CAF1 E846040000          <1>       call   load_FAT_sub_directory
4365 0000CAF6 72D3                <1>       jc     short loc_createfile_anc_retn
4366                              <1>
4367                              <1> pass_createfile_add_new_subdir_cluster:
4368                              <1>       ;movzx eax, word [esi+LD_BPB+BytesPerSec]
4369 0000CAF8 0FB705[BC8D0100]    <1>       movzx  eax, word [createfile_BytesPerSec] ; 23/03/2016
4370 0000CAFF F7E1                <1>       mul    ecx ; ecx = directory buffer sector count
4371 0000CB01 89C1                <1>       mov    ecx, eax
4372 0000CB03 C1E902              <1>       shr    ecx, 2 ; dword count
4373 0000CB06 29C0                <1>       sub    eax, eax ; 0
4374 0000CB08 F3AB                <1>       rep    stosd
4375                              <1>       ;
4376 0000CB0A C605[7C890100]02    <1>       mov    byte [DirBuff_ValidData], 2
4377 0000CB11 E890EFFFFF          <1>       call   save_directory_buffer
4378 0000CB16 72B3                <1>       jc     short loc_createfile_anc_retn
4379                              <1>
4380                              <1> loc_createfile_save_added_subdir_cluster:
4381 0000CB18 A1[A88D0100]        <1>       mov    eax, [createfile_LastDirCluster]
4382 0000CB1D 8B0D[A48D0100]      <1>       mov    ecx, [createfile_FFCluster]
4383 0000CB23 E858050000          <1>       call   update_cluster
4384 0000CB28 7304                <1>       jnc    short loc_createfile_save_fat_buffer_0
4385 0000CB2A 09C0                <1>       or     eax, eax ; EAX = 0 -> cluster value is 0 or eocc
4386 0000CB2C 751A                <1>       jnz    short loc_createfile_save_fat_buffer_stc_retn
4387                              <1>
4388                              <1> loc_createfile_save_fat_buffer_0:
4389 0000CB2E A1[A48D0100]        <1>       mov    eax, [createfile_FFCluster]
4390 0000CB33 A3[A88D0100]        <1>       mov    [createfile_LastDirCluster], eax
4391 0000CB38 B9FFFFFF0F          <1>       mov    ecx, 0FFFFFFFh ; 28 bit
4392 0000CB3D E83E050000          <1>       call   update_cluster
4393 0000CB42 7306                <1>       jnc    short loc_createfile_save_fat_buffer_1
4394 0000CB44 09C0                <1>       or     eax, eax ; Was it free cluster
4395 0000CB46 7402                <1>       jz     short loc_createfile_save_fat_buffer_1
4396                              <1>
4397                              <1> loc_createfile_save_fat_buffer_stc_retn:
4398 0000CB48 F9                  <1>       stc
4399                              <1> loc_createfile_save_fat_buffer_retn:
4400                              <1> loc_createfile_gffc_2_stc_retn:
4401 0000CB49 C3                  <1>       retn
4402                              <1>
4403                              <1> loc_createfile_save_fat_buffer_1:
4404                              <1>       ; byte [FAT_BuffValidData] = 2
4405 0000CB4A E8EE070000          <1>       call   save_fat_buffer
4406 0000CB4F 72F8                <1>       jc     short loc_createfile_save_fat_buffer_retn
4407                              <1>
4408 0000CB51 803D[72890100]01    <1>       cmp    byte [FAT_ClusterCounter], 1
4409 0000CB58 7222                <1>       jb     short loc_createfile_save_fat_buffer_2
4410                              <1>
4411                              <1>       ; ESI = Logical DOS Drive Description Table address
4412 0000CB5A A1[72890100]        <1>       mov    eax, [FAT_ClusterCounter]
4413                              <1>
4414 0000CB5F C605[72890100]00    <1>       mov    byte [FAT_ClusterCounter], 0 ; 21/03/2016
4415                              <1>
4416 0000CB66 66BB01FF            <1>       mov    bx, 0FF01h ; add free clusters
4417 0000CB6A E863080000          <1>       call   calculate_fat_freespace
4418                              <1>
4419                              <1>       ;inc   eax ; 0FFFFFFFFh -> 0 ; recalculation is needed!
4420                              <1>       ;jnz   short loc_createfile_save_fat_buffer_2
4421                              <1>
4422                              <1>       ; ecx > 0 -> Recalculation is needed
4423 0000CB6F 09C9                <1>       or     ecx, ecx
4424 0000CB71 7409                <1>       jz     short loc_createfile_save_fat_buffer_2
4425                              <1>
4426 0000CB73 66BB00FF            <1>       mov    bx, 0FF00h ; ; recalculate free space
4427 0000CB77 E856080000          <1>       call   calculate_fat_freespace
4428                              <1>
4429                              <1> loc_createfile_save_fat_buffer_2:
4430                              <1>       ;call  update_parent_dir_lmdt
4431                              <1>
4432                              <1> loc_createfile_gffc_2:
4433 0000CB7C E82C040000          <1>       call   get_first_free_cluster
4434 0000CB81 72C6                <1>       jc     short loc_createfile_gffc_2_stc_retn
4435                              <1>
4436 0000CB83 A3[A48D0100]        <1>       mov    [createfile_FFCluster], eax
4437                              <1>
4438 0000CB88 A1[A88D0100]        <1>       mov    eax, [createfile_LastDirCluster]
4439                              <1>
4440 0000CB8D E8AA030000          <1>       call   load_FAT_sub_directory
4441 0000CB92 72B5                <1>       jc     short loc_createfile_gffc_2_stc_retn
4442                              <1>
4443 0000CB94 BF00000800          <1>       mov    edi, Directory_Buffer
4444                              <1>
4445 0000CB99 6629DB              <1>       sub    bx, bx ; directory entry index/number = 0
```

```
4446                                    <1>
4447 0000CB9C 56                        <1>       push   esi ; * ; 23/03/2016
4448                                    <1>
4449                                    <1> loc_createfile_set_ff_dir_entry:
4450 0000CB9D 66891D[B68D0100]          <1>       mov    [createfile_DirIndex], bx
4451                                    <1>
4452                                    <1>         ; EDI = Directory entry address
4453 0000CBA4 8B35[988D0100]            <1>       mov    esi, [createfile_Name_Offset]
4454 0000CBAA A1[A48D0100]              <1>       mov    eax, [createfile_FFCluster]
4455 0000CBAF A3[AC8D0100]              <1>       mov    [createfile_Cluster], eax ; 24/03/2016
4456 0000CBB4 B5FF                      <1>       mov    ch, 0FFh
4457 0000CBB6 8A0D[B48D0100]            <1>         mov cl, [createfile_attrib] ; file attributes
4458                                    <1>       ; CH > 0 -> File size is in [EBX]
4459 0000CBBC BB[A08D0100]              <1>       mov    ebx, createfile_size
4460                                    <1>
4461 0000CBC1 E803EEFFFF                <1>       call   make_directory_entry
4462                                    <1>
4463 0000CBC6 5E                        <1>       pop    esi ; * ; ESI = Logical Dos Drv Desc. Table address
4464                                    <1>
4465 0000CBC7 C605[7C890100]02          <1>       mov    byte [DirBuff_ValidData], 2
4466 0000CBCE E8D3EEFFFF                <1>       call   save_directory_buffer
4467 0000CBD3 7221                      <1>       jc     short loc_createfile_set_ff_dir_entry_retn
4468                                    <1>
4469 0000CBD5 C605[BF8D0100]01          <1>       mov    byte [createfile_UpdatePDir], 1 ; 31/03/2016
4470                                    <1>
4471                                    <1> loc_createfile_get_set_write_file_cluster:
4472 0000CBDC A1[A08D0100]              <1>       mov    eax, [createfile_size]
4473 0000CBE1 09C0                      <1>       or     eax, eax
4474 0000CBE3 7570                      <1>       jnz    short loc_createfile_get_set_wfc_cont
4475 0000CBE5 40                        <1>       inc    eax
4476                                    <1>       ; 23/03/2016
4477 0000CBE6 0FB61D[B58D0100]          <1>       movzx  ebx, byte [createfile_SecPerClust]
4478                                    <1>       ;movzx ecx, word [esi+LD_BPB+BytesPerSec] ; 512
4479 0000CBED 0FB70D[BC8D0100]          <1>         movzx   ecx, word [createfile_BytesPerSec] ; 512
4480 0000CBF4 EB7C                      <1>       jmp    loc_createfile_set_cluster_count
4481                                    <1>
4482                                    <1> loc_createfile_set_ff_dir_entry_retn:
4483 0000CBF6 C3                        <1>       retn
4484                                    <1>
4485                                    <1> loc_createfile_write_fcluster_to_disk:
4486 0000CBF7 034668                    <1>       add    eax, [esi+LD_DATABegin] ; convert to physical address
4487 0000CBFA BB00000700                <1>       mov    ebx, Cluster_Buffer
4488                                    <1>       ; ESI = Logical DOS Drv. Desc. Tbl. address
4489                                    <1>       ; EAX = Disk address
4490                                    <1>       ; EBX = Sector Buffer
4491                                    <1>       ; ECX = sectors per cluster
4492 0000CBFF E80A570000                <1>       call   disk_write
4493 0000CC04 7211                      <1>       jc     short loc_createfile_dsk_wr_err
4494                                    <1>
4495                                    <1> loc_createfile_update_fat_cluster:
4496                                    <1>       ; 21/03/2016
4497 0000CC06 803D[BE8D0100]00          <1>       cmp    byte [createfile_wfc], 0
4498 0000CC0D 7712                      <1>       ja     short loc_createfile_update_fat_cluster_n1
4499                                    <1>
4500 0000CC0F FE05[BE8D0100]            <1>       inc    byte [createfile_wfc] ; 1
4501 0000CC15 EB24                      <1>       jmp    short loc_createfile_update_fat_cluster_n2
4502                                    <1>
4503                                    <1> loc_createfile_dsk_wr_err:
4504                                    <1>       ; 16/10/2016 (1Dh -> 18)
4505                                    <1>       ; 23/03/2016
4506 0000CC17 B812000000                <1>       mov    eax, 18 ; Drive not ready or write error !
4507 0000CC1C E9BD000000                <1>       jmp    loc_createfile_stc_retn
4508                                    <1>
4509                                    <1> loc_createfile_update_fat_cluster_n1:
4510 0000CC21 A1[B08D0100]              <1>       mov    eax, [createfile_PCluster]
4511 0000CC26 8B0D[AC8D0100]            <1>       mov    ecx, [createfile_Cluster]
4512 0000CC2C E84F040000                <1>       call   update_cluster
4513 0000CC31 7308                      <1>       jnc    short loc_createfile_update_fat_cluster_n2
4514 0000CC33 09C0                      <1>       or     eax, eax ; EAX = 0 -> cluster value is 0 or eocc
4515 0000CC35 0F85A3000000              <1>       jnz    loc_createfile_stc_retn
4516                                    <1>
4517                                    <1> loc_createfile_update_fat_cluster_n2:
4518 0000CC3B A1[AC8D0100]              <1>         mov  eax, [createfile_Cluster]
4519 0000CC40 B9FFFFFF0F                <1>       mov    ecx, 0FFFFFFFh
4520 0000CC45 E836040000                <1>       call   update_cluster
4521 0000CC4A 734E                      <1>       jnc    short loc_createfile_save_fat_buffer_3
4522 0000CC4C 09C0                      <1>       or     eax, eax ; EAX = 0 -> cluster value is 0 or eocc
4523 0000CC4E 744A                      <1>       jz     short loc_createfile_save_fat_buffer_3
4524                                    <1>
4525                                    <1> loc_createfile_upd_fat_fcluster_stc_retn:
4526 0000CC50 E989000000                <1>       jmp    loc_createfile_stc_retn
4527                                    <1>
4528                                    <1> loc_createfile_get_set_wfc_cont:
4529                                    <1>       ;movzx ecx, word [esi+LD_BPB+BytesPerSec] ; 512
4530 0000CC55 0FB70D[BC8D0100]          <1>       movzx  ecx, word [createfile_BytesPerSec] ; 512
4531 0000CC5C 01C8                      <1>       add    eax, ecx
4532 0000CC5E 48                        <1>       dec    eax  ; add eax, 511
4533 0000CC5F 29D2                      <1>       sub    edx, edx
4534 0000CC61 F7F1                      <1>       div    ecx
4535 0000CC63 0FB61D[B58D0100]          <1>       movzx  ebx, byte [createfile_SecPerClust]
4536 0000CC6A 01D8                      <1>       add    eax, ebx
4537 0000CC6C 48                        <1>       dec    eax  ; add eax, SecPerClust - 1
4538 0000CC6D 6631D2                    <1>       xor    dx, dx
4539 0000CC70 F7F3                      <1>       div    ebx
4540                                    <1>
4541                                    <1> loc_createfile_set_cluster_count:
4542 0000CC72 A3[B88D0100]              <1>       mov    [createfile_CCount], eax
4543                                    <1>
4544 0000CC77 BF00000700                <1>       mov    edi, Cluster_Buffer
4545 0000CC7C 89C8                      <1>       mov    eax, ecx ; Bytes per Sector
4546 0000CC7E F7E3                      <1>       mul    ebx ; Sectors per Cluster
4547                                    <1>       ; EAX = Bytes per Cluster
4548 0000CC80 89C1                      <1>       mov    ecx, eax
4549 0000CC82 C1E902                    <1>       shr    ecx, 2 ; dword count
4550 0000CC85 31C0                      <1>       xor    eax, eax
```

```
4551 0000CC87 F3AB               <1>       rep    stosd ; clear cluster buffer
4552                             <1>
4553 0000CC89 A1[AC8D0100]       <1>       mov    eax, [createfile_Cluster] ; 24/03/2016
4554                             <1>
4555 0000CC8E 89D9               <1>       mov    ecx, ebx
4556                             <1>
4557                             <1> loc_createfile_get_set_wf_fclust_cont:
4558 0000CC90 83E802             <1>       sub    eax, 2
4559 0000CC93 F7E1               <1>       mul    ecx
4560                             <1>       ; EAX = Logical DOS disk address (offset)
4561 0000CC95 E95DFFFFFF         <1>         jmp     loc_createfile_write_fcluster_to_disk
4562                             <1>
4563                             <1> loc_createfile_save_fat_buffer_3:
4564                             <1>       ; byte [FAT_BuffValidData] = 2
4565 0000CC9A E89E060000         <1>       call   save_fat_buffer
4566 0000CC9F 723D               <1>       jc     loc_createfile_stc_retn
4567                             <1>
4568                             <1>       ; 21/03/2016
4569 0000CCA1 803D[72890100]01   <1>       cmp    byte [FAT_ClusterCounter], 1
4570 0000CCA8 721B               <1>       jb     short loc_createfile_save_fat_buffer_4
4571                             <1>
4572                             <1>       ; ESI = Logical DOS Drive Description Table address
4573 0000CCAA A1[72890100]       <1>       mov    eax, [FAT_ClusterCounter]
4574 0000CCAF 66BB01FF           <1>       mov    bx, 0FF01h ; add free clusters
4575 0000CCB3 E81A070000         <1>       call   calculate_fat_freespace
4576                             <1>
4577                             <1>       ;inc   eax ; 0FFFFFFFFh -> 0 ; recalculation is needed!
4578                             <1>       ;jnz   short loc_createfile_save_fat_buffer_4
4579                             <1>
4580                             <1>       ; ecx > 0 -> Recalculation is needed
4581 0000CCB8 09C9               <1>       or     ecx, ecx
4582 0000CCBA 7409               <1>       jz     short loc_createfile_save_fat_buffer_4
4583                             <1>
4584 0000CCBC 66BB00FF           <1>       mov    bx, 0FF00h ; ; recalculate free space
4585 0000CCC0 E80D070000         <1>       call   calculate_fat_freespace
4586                             <1>
4587                             <1> loc_createfile_save_fat_buffer_4:
4588 0000CCC5 FF0D[B88D0100]     <1>       dec    dword [createfile_CCount]
4589                             <1>       ;jz    short loc_createfile_upd_dir_modif_date_time
4590 0000CCCB 743F               <1>       jz     short loc_createfile_stc_retn_cc ; 31/03/2016
4591                             <1>
4592                             <1> loc_createfile_get_set_write_next_cluster:
4593 0000CCCD E8DB020000         <1>       call   get_first_free_cluster
4594 0000CCD2 720A               <1>       jc     short loc_createfile_stc_retn
4595                             <1>
4596                             <1> loc_createfile_get_set_write_next_cluster_1:
4597 0000CCD4 83F8FF             <1>       cmp    eax, 0FFFFFFFFh
4598 0000CCD7 7213               <1>       jb     short loc_createfile_get_set_write_next_cluster_2
4599                             <1>
4600                             <1> loc_createfile_wnc_insufficient_disk_space:
4601 0000CCD9 B827000000         <1>       mov    eax, 27h ; Insufficient disk space
4602                             <1>
4603                             <1> loc_createfile_stc_retn:
4604 0000CCDE 803D[BE8D0100]01   <1>       cmp    byte [createfile_wfc], 1
4605 0000CCE5 7324               <1>       jnb    short loc_createfile_err_retn
4606 0000CCE7 C3                 <1>       retn
4607                             <1>
4608                             <1> loc_createfile_wnc_inv_format_retn:
4609                             <1>       ;mov   eax, 28
4610 0000CCE8 B01C               <1>       mov    al, 28 ; Invalid format
4611 0000CCEA EBF2               <1>       jmp    short loc_createfile_stc_retn
4612                             <1>
4613                             <1> loc_createfile_get_set_write_next_cluster_2:
4614 0000CCEC 83F802             <1>       cmp    eax, 2
4615 0000CCEF 72F7               <1>       jb     short loc_createfile_wnc_inv_format_retn
4616                             <1>
4617                             <1> loc_createfile_get_set_write_next_cluster_3:
4618 0000CCF1 8B0D[AC8D0100]     <1>       mov    ecx, [createfile_Cluster]
4619 0000CCF7 A3[AC8D0100]       <1>       mov    [createfile_Cluster], eax
4620 0000CCFC 890D[B08D0100]     <1>       mov    [createfile_PCluster], ecx
4621 0000CD02 0FB60D[B58D0100]   <1>       movzx  ecx, byte [createfile_SecPerClust]
4622 0000CD09 EB85               <1>       jmp    short loc_createfile_get_set_wf_fclust_cont
4623                             <1>
4624                             <1> loc_createfile_err_retn:
4625 0000CD0B F9                 <1>       stc
4626                             <1>
4627                             <1> ;loc_createfile_upd_dir_modif_date_time:
4628                             <1> loc_createfile_stc_retn_cc: ; 31/03/2016
4629 0000CD0C 9C                 <1>       pushf  ; cpu is here for an error return or completion
4630 0000CD0D 50                 <1>       push   eax ; error code if cf = 1
4631                             <1>
4632                             <1>       ;call  update_parent_dir_lmdt
4633                             <1>
4634                             <1> ;loc_createfile_stc_retn_cc:
4635 0000CD0E A1[72890100]       <1>       mov    eax, [FAT_ClusterCounter]
4636 0000CD13 09C0               <1>       or     eax, eax
4637 0000CD15 741A               <1>       jz     short loc_createfile_stc_retn_pop_eax
4638 0000CD17 8A3D[56820100]     <1>       mov    bh, [Current_Drv]
4639 0000CD1D B301               <1>       mov    bl, 01h ; BL = 1 -> add clusters
4640                             <1>       ; NOTE: EAX value will be added to Free Cluster Count
4641                             <1>       ; (If EAX value is negative, Free Cluster Count will be decreased)
4642 0000CD1F E8AE060000         <1>       call   calculate_fat_freespace
4643                             <1>         ; ESI = Logical DOS Drive Description Table Address
4644                             <1>         ;jc  short loc_createfile_stc_retn_pop_eax_cf
4645 0000CD24 21C9               <1>       and    ecx, ecx ; cx = 0 -> valid free sector count
4646 0000CD26 7409               <1>       jz     short loc_createfile_stc_retn_pop_eax
4647                             <1>
4648                             <1> loc_createfile_stc_retn_recalc_FAT_freespace:
4649 0000CD28 66BB00FF           <1>       mov    bx, 0FF00h ; bh = 0FFh ->
4650                             <1>       ; ESI = Logical DOS Drv DT Addr
4651                             <1>       ; BL = 0 -> Recalculate
4652 0000CD2C E8A1060000         <1>       call   calculate_fat_freespace
4653                             <1>
4654                             <1> loc_createfile_stc_retn_pop_eax:
4655 0000CD31 58                 <1>       pop    eax
```

```
4656 0000CD32 9D                <1>       popf
4657 0000CD33 7218              <1>       jc     short loc_createfile_retn
4658                            <1>
4659                            <1> loc_createfile_retn_fcluster:
4660 0000CD35 A1[A48D0100]      <1>       mov    eax, [createfile_FFCluster]
4661 0000CD3A BB[A08D0100]      <1>       mov    ebx, createfile_size
4662                            <1>       ;movzx ecx, byte [esi+LD_BPB+SecPerClust]
4663 0000CD3F 0FB60D[B58D0100]  <1>       movzx  ecx, byte [createfile_SecPerClust] ; 23/03/2016
4664 0000CD46 0FB715[B68D0100]  <1>       movzx  edx, word [createfile_DirIndex]
4665                            <1>
4666                            <1> loc_createfile_retn:
4667 0000CD4D C3                <1>       retn
4668                            <1>
4669                            <1> create_fs_file:
4670                            <1>       ; temporary (21/03/2016)
4671 0000CD4E C3                <1>       retn
4672                            <1>
4673                            <1> delete_fs_file:
4674                            <1>       ; temporary (28/02/2016)
4675 0000CD4F C3                <1>       retn
4676                            <1>
4677                            <1> rename_fs_file_or_directory:
4678 0000CD50 C3                <1>       retn
4679                            <1>
4680                            <1> make_fs_directory:
4681                            <1>       ; temporary (21/02/2016)
4682 0000CD51 C3                <1>       retn
4683                            <1>
4684                            <1> add_new_fs_section:
4685                            <1>       ; temporary (11/03/2016)
4686 0000CD52 C3                <1>       retn
4687                            <1>
4688                            <1> delete_fs_directory_entry:
4689                            <1>       ; temporary (11/03/2016)
4690 0000CD53 C3                <1>       retn
4691                            <1>
4692                            <1> csftdf2_read_fs_file_sectors:
4693                            <1>       ; temporary (19/03/2016)
4694 0000CD54 C3                <1>       retn
4695                            <1>
4696                            <1> csftdf2_write_fs_file_sectors:
4697                            <1>       ; temporary (19/03/2016)
4698 0000CD55 C3                <1>       retn
3083                                %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>       ;
  32                            <1>       ; (Modified registers: EAX, ECX, EBX, EDX)
  33                            <1>
  34 0000CD56 A3[66890100]      <1>       mov    [FAT_CurrentCluster], eax
  35                            <1> check_next_cluster_fat_type:
  36 0000CD5B 29D2              <1>       sub    edx, edx ; 0
  37 0000CD5D 807E0302          <1>         cmp     byte [esi+LD_FATType], 2
  38 0000CD61 7250              <1>       jb     short get_FAT12_next_cluster
  39 0000CD63 0F87AF000000      <1>         ja      get_FAT32_next_cluster
  40                            <1> get_FAT16_next_cluster:
  41 0000CD69 BB00030000        <1>       mov    ebx, 300h ;768
  42 0000CD6E F7F3              <1>       div    ebx
  43                            <1>       ; EAX = Count of 3 FAT sectors
  44                            <1>       ; EDX = Cluster Offset (< 768)
  45 0000CD70 66D1E2            <1>       shl    dx, 1 ; Multiply by 2
  46 0000CD73 89D3              <1>       mov    ebx, edx ; Byte Offset
  47 0000CD75 81C3001C0900      <1>       add    ebx, FAT_Buffer
  48 0000CD7B 66BA0300          <1>       mov    dx, 3
  49 0000CD7F F7E2              <1>       mul    edx
  50                            <1>       ; EAX = FAT Sector (<= 256)
  51                            <1>       ; EDX = 0
  52 0000CD81 8A0E              <1>       mov    cl, [esi+LD_Name]
  53 0000CD83 803D[6A890100]00  <1>       cmp    byte [FAT_BuffValidData], 0
  54 0000CD8A 0F86CC000000      <1>         jna     load_FAT_sectors0
  55 0000CD90 3A0D[6B890100]    <1>       cmp    cl, [FAT_BuffDrvName]
  56 0000CD96 0F85C0000000      <1>         jne     load_FAT_sectors0
  57 0000CD9C 3B05[6E890100]    <1>       cmp    eax, [FAT_BuffSector]
  58 0000CDA2 0F85BA000000      <1>         jne     load_FAT_sectors1
  59                            <1>       ;movzx eax, word [ebx]
  60 0000CDA8 668B03            <1>       mov    ax, [ebx]
  61                            <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
0000CDAB 6683F8F7               <1>       cmp    ax, 0FFF7h
0000CDAF 725A                   <1>       jb     short loc_pass_gnc_FAT16_eoc_check
                                <1>       ; ax >= FFF7h (cluster 0002h to FFF6h is valid, in use)
0000CDB1 EB56                   <1>       jmp    short loc_pass_gnc_FAT16_eoc_check_xor_eax
                                <1>
                                <1> get_FAT12_next_cluster:
0000CDB3 BB00040000             <1>       mov    ebx, 400h ;1024
0000CDB8 F7F3                   <1>       div    ebx
                                <1>       ; EAX = Count of 3 FAT sectors
                                <1>       ; EDX = Cluster Offset (< 1024)
0000CDBA 6650                   <1>       push   ax
0000CDBC 66B80300               <1>       mov    ax, 3
0000CDC0 66F7E2                 <1>       mul    dx     ; Multiply by 3
0000CDC3 66D1E8                 <1>       shr    ax, 1  ; Divide by 2
0000CDC6 6689C3                 <1>         mov  bx, ax       ; Byte Offset
0000CDC9 81C3001C0900           <1>       add    ebx, FAT_Buffer
0000CDCF 6658                   <1>       pop    ax
0000CDD1 66BA0300               <1>       mov    dx, 3
0000CDD5 F7E2                   <1>       mul    edx
                                <1>       ; EAX = FAT Sector (<= 12)
                                <1>       ; EDX = 0
0000CDD7 8A0E                   <1>       mov    cl, [esi+LD_Name]
0000CDD9 803D[6A890100]00       <1>       cmp    byte [FAT_BuffValidData], 0
0000CDE0 767A                   <1>       jna    short load_FAT_sectors0
0000CDE2 3A0D[6B890100]         <1>       cmp    cl, [FAT_BuffDrvName]
0000CDE8 7572                   <1>       jne    short load_FAT_sectors0
0000CDEA 3B05[6E890100]         <1>       cmp    eax, [FAT_BuffSector]
0000CDF0 7570                   <1>       jne    short load_FAT_sectors1
0000CDF2 A1[66890100]           <1>       mov    eax, [FAT_CurrentCluster]
0000CDF7 66D1E8                 <1>       shr    ax, 1
                                <1>       ;movzx eax, word [ebx]
0000CDFA 668B03                 <1>       mov    ax, [ebx]
0000CDFD 7314                   <1>       jnc    short get_FAT12_nc_even
0000CDFF 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
0000CE03 663DF70F               <1>       cmp    ax, 0FF7h
0000CE07 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:
0000CE09 31C0                   <1>       xor    eax, eax ; 0
                                <1> loc_pass_gnc_FAT16_eoc_check:
                                <1> loc_pass_gnc_FAT32_eoc_check:
0000CE0B 8B0D[66890100]         <1>       mov    ecx, [FAT_CurrentCluster]
0000CE11 F5                     <1>       cmc
0000CE12 C3                     <1>       retn
                                <1>
                                <1> get_FAT12_nc_even:
0000CE13 80E40F                 <1>       and    ah, 0Fh
0000CE16 EBEB                   <1>       jmp    short loc_gnc_fat12_eoc_check
                                <1>
                                <1> get_FAT32_next_cluster:
0000CE18 BB80010000             <1>       mov    ebx, 180h ;384
0000CE1D F7F3                   <1>       div    ebx
                                <1>       ; EAX = Count of 3 FAT sectors
                                <1>       ; EDX = Cluster Offset (< 384)
0000CE1F 66C1E202               <1>       shl    dx, 2  ; Multiply by 4
0000CE23 89D3                   <1>       mov    ebx, edx ; Byte Offset
0000CE25 81C3001C0900           <1>       add    ebx, FAT_Buffer
0000CE2B 66BA0300               <1>       mov    dx, 3
0000CE2F 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
0000CE31 8A0E                   <1>       mov    cl, [esi+LD_Name]
0000CE33 803D[6A890100]00       <1>       cmp    byte [FAT_BuffValidData], 0
0000CE3A 7620                   <1>       jna    short load_FAT_sectors0
0000CE3C 3A0D[6B890100]         <1>       cmp    cl, [FAT_BuffDrvName]
0000CE42 7518                   <1>       jne    short load_FAT_sectors0
0000CE44 3B05[6E890100]         <1>       cmp    eax, [FAT_BuffSector] ; 0, 3, 6, 9 ...
0000CE4A 7516                   <1>       jne    short load_FAT_sectors1
0000CE4C 8B03                   <1>       mov    eax, [ebx]
0000CE4E 25FFFFFF0F             <1>       and    eax, 0FFFFFFFh ; 28 bit Cluster
0000CE53 3DF7FFFF0F             <1>       cmp    eax, 0FFFFFF7h
0000CE58 72B1                   <1>       jb     short loc_pass_gnc_FAT32_eoc_check
                                <1>       ; eax >= FFFFFF7h (cluster 0002h to FFFFFF6h is valid)
0000CE5A EBAD                   <1>       jmp    short loc_pass_gnc_FAT16_eoc_check_xor_eax
                                <1>
                                <1> load_FAT_sectors0:
0000CE5C 880D[6B890100]         <1>       mov    [FAT_BuffDrvName], cl
                                <1> load_FAT_sectors1:
0000CE62 A3[6E890100]           <1>       mov    [FAT_BuffSector], eax
0000CE67 89C3                   <1>       mov    ebx, eax
0000CE69 034660                 <1>         add    eax, [esi+LD_FATBegin]
0000CE6C 807E0302               <1>       cmp    byte [esi+LD_FATType], 2
0000CE70 7706                   <1>         ja     short load_FAT_sectors3
0000CE72 0FB74E1C               <1>       movzx  ecx, word [esi+LD_BPB+BPB_FATSz16]
0000CE76 EB03                   <1>       jmp    short load_FAT_sectors4
                                <1> load_FAT_sectors3:
0000CE78 8B4E2A                 <1>       mov    ecx, [esi+LD_BPB+BPB_FATSz32]
                                <1> load_FAT_sectors4:
0000CE7B 29D9                   <1>       sub    ecx, ebx ; [FAT_BuffSector]
0000CE7D 83F903                 <1>         cmp    ecx, 3
```

```
0000CE80 7605                <1>         jna     short load_FAT_sectors5
0000CE82 B903000000          <1>       mov   ecx, 3
                             <1> load_FAT_sectors5:
0000CE87 BB001C0900          <1>       mov   ebx, FAT_Buffer
0000CE8C E88C540000          <1>       call  disk_read
0000CE91 730D                <1>       jnc   short load_FAT_sectors_ok
                             <1>       ; 15/10/2016 (15h -> 17)
                             <1>       ; 23/03/2016 (15h)
0000CE93 B811000000          <1>       mov   eax, 17 ; Drive not ready or read error
0000CE98 C605[6A890100]00    <1>       mov   byte [FAT_BuffValidData], 0
0000CE9F C3                  <1>       retn
                             <1> load_FAT_sectors_ok:
0000CEA0 C605[6A890100]01    <1>       mov   byte [FAT_BuffValidData], 1
0000CEA7 A1[66890100]        <1>       mov   eax, [FAT_CurrentCluster]
0000CEAC 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>
0000CEB1 8A1E                <1>       mov   bl, [esi+LD_Name]
0000CEB3 8A7E03              <1>       mov   bh, [esi+LD_FATType]
                             <1>
                             <1>       ;mov  [DirBuff_DRV], bl
                             <1>       ;mov  [DirBuff_FATType], bh
0000CEB6 66891D[7A890100]    <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
0000CEBD 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>
0000CEC1 6681FA0002          <1>       cmp   dx, 512 ; Number of Root Dir Entries
0000CEC6 7414                <1>       je    short lrd_mov_ecx_32
0000CEC8 89D0                <1>       mov   eax, edx
                             <1>       ; 23/10/2016
0000CECA 89C1                <1>       mov   ecx, eax
0000CECC 6683C10F            <1>       add   cx, 15 ; round up
0000CED0 66C1E904            <1>       shr   cx, 4  ; 16 entries per sector (512/32)
                             <1>       ; ecx = Root directory size in sectors
0000CED4 66C1E005            <1>       shl   ax, 5 ; Root directory size in bytes
0000CED8 664A                <1>       dec   dx    ; Last entry number of root dir
                             <1>       ; cx = Dir Buffer sector count
0000CEDA EB0B                <1>       jmp   short lrd_check_dir_buffer
                             <1>
                             <1> lrd_mov_ecx_32:
0000CEDC B920000000          <1>       mov   ecx, 32
0000CEE1 664A                <1>       dec   dx ; 511
0000CEE3 66B80040            <1>       mov   ax, 32*512
                             <1>
                             <1> lrd_check_dir_buffer:
0000CEE7 29DB                <1>       sub   ebx, ebx ; 0
0000CEE9 881D[7C890100]      <1>       mov   [DirBuff_ValidData], bl ; 0
0000CEEF 668915[7F890100]    <1>       mov   [DirBuff_LastEntry], dx
0000CEF6 891D[81890100]      <1>       mov   [DirBuff_Cluster], ebx ; 0
0000CEFC 66A3[85890100]      <1>       mov   [DirBuffer_Size], ax
                             <1>
0000CF02 8B4664              <1>       mov   eax, [esi+LD_ROOTBegin]
                             <1> read_directory:
0000CF05 BB00000800          <1>       mov   ebx, Directory_Buffer
0000CF0A 51                  <1>       push  ecx ; Directory buffer sector count
0000CF0B 53                  <1>       push  ebx
0000CF0C E80C540000          <1>       call  disk_read
0000CF11 5B                  <1>       pop   ebx
0000CF12 720B                <1>       jc    short load_DirBuff_error
                             <1>
                             <1> validate_DirBuff_and_return:
0000CF14 59                  <1>       pop   ecx ; Number of loaded sectors
0000CF15 C605[7C890100]01    <1>       mov   byte [DirBuff_ValidData], 1
0000CF1C 31C0                <1>       xor   eax, eax ; 0 = no error
0000CF1E C3                  <1>       retn
                             <1>
                             <1> load_DirBuff_error:
0000CF1F 89C8                <1>       mov   eax, ecx ; remaining sectors
0000CF21 59                  <1>       pop   ecx ; sector count
0000CF22 29C1                <1>       sub   ecx, eax ; Number of loaded sectors
                             <1>       ; 15/10/2016 (15h -> 17)
0000CF24 B811000000          <1>       mov   eax, 17 ; DRV NOT READY OR READ ERROR !
0000CF29 F9                  <1>       stc
0000CF2A 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>
0000CF2B 8A1E               <1>       mov   bl, [esi+LD_Name]
0000CF2D 8A7E03             <1>       mov   bh, [esi+LD_FATType]
                            <1>
                            <1>       ;mov  [DirBuff_DRV], bl
                            <1>       ;mov  [DirBuff_FATType], bh
0000CF30 66891D[7A890100]   <1>       mov   [DirBuff_DRV], bx
                            <1>
                            <1> load_FAT32_root_dir0:
0000CF37 8B4632             <1>       mov   eax, [esi+LD_BPB+FAT32_RootFClust]
0000CF3A 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>
0000CF3C 8A1E               <1>       mov   bl, [esi+LD_Name]
0000CF3E 8A7E03             <1>       mov   bh, [esi+LD_FATType]
                            <1>
                            <1>       ;mov  [DirBuff_DRV], bl
                            <1>       ;mov  [DirBuff_FATType], bh
0000CF41 66891D[7A890100]   <1>       mov   [DirBuff_DRV], bx
                            <1>
                            <1> load_FAT_sub_dir0:
0000CF48 0FB64E13           <1>       movzx ecx, byte [esi+LD_BPB+SecPerClust]
                            <1>
0000CF4C 882D[7C890100]     <1>       mov   [DirBuff_ValidData], ch ; 0
0000CF52 A3[81890100]       <1>       mov   [DirBuff_Cluster], eax
                            <1>
0000CF57 0FB74611           <1>       movzx eax, word [esi+LD_BPB+BytesPerSec]
0000CF5B F7E1               <1>       mul   ecx
0000CF5D C1E805             <1>       shr   eax, 5 ; directory entry count (dir size / 32)
0000CF60 6648               <1>       dec   ax ; last entry
0000CF62 66A3[7F890100]     <1>       mov   [DirBuff_LastEntry], ax
                            <1>
0000CF68 A1[81890100]       <1>       mov   eax, [DirBuff_Cluster]
0000CF6D 83E802             <1>       sub   eax, 2
0000CF70 F7E1               <1>       mul   ecx
0000CF72 034668             <1>       add   eax, [esi+LD_DATABegin]
                            <1>       ; ecx = sector per cluster (dir buffer size = 32 sectors)
0000CF75 EB8E               <1>       jmp   short read_directory
                            <1>
                            <1> ; DRV_FS.ASM
                            <1>
                            <1> load_current_FS_directory:
0000CF77 C3                 <1>       retn
                            <1> load_FS_root_directory:
0000CF78 C3                 <1>       retn
                            <1> load_FS_sub_directory:
0000CF79 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>
0000CF7A 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
```

```
0000CF7E 807E0300            <1>       cmp    byte [esi+LD_FATType], 0
0000CF82 761C                <1>       jna    short read_fs_cluster
                             <1>
                             <1> read_fat_file_sectors: ; 18/03/2016
0000CF84 83E802              <1>       sub    eax, 2 ; Beginning cluster number is always 2
0000CF87 0FB65613            <1>       movzx  edx, byte [esi+LD_BPB+BPB_SecPerClust] ; 18/03/2016
0000CF8B F7E2                <1>       mul    edx
0000CF8D 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>
0000CF90 E888530000          <1>       call   disk_read
0000CF95 7306                <1>       jnc    short rclust_retn
                             <1>
                             <1>       ; 15/10/2016 (15h -> 17)
0000CF97 B811000000          <1>       mov    eax, 17 ; Drive not ready or read error !
0000CF9C C3                  <1>       retn
                             <1>
                             <1> rclust_retn:
0000CF9D 29C0                <1>       sub    eax, eax ; 0
0000CF9F 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)
0000CFA0 B980000000          <1>       mov    ecx, 128 ; maximum count of sectors (before eof)
0000CFA5 E801000000          <1>       call   read_fs_sectors
0000CFAA C3                  <1>       retn
                             <1>
                             <1> read_fs_sectors:
                             <1>       ; 15/02/2016 (TRDOS 386 =  TRDOS v2.0)
0000CFAB F9                  <1>       stc
0000CFAC 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>
0000CFAD 8B4678              <1>       mov    eax, [esi+LD_Clusters]
0000CFB0 40                  <1>       inc    eax ; add eax, 1
0000CFB1 A3[048C0100]        <1>       mov    [gffc_last_free_cluster], eax
                             <1>
0000CFB6 31DB                <1>       xor    ebx, ebx ; 0 ; 02/03/2016
                             <1>
0000CFB8 807E0302            <1>       cmp    byte [esi+LD_FATType], 2
0000CFBC 760E                <1>       jna    short loc_gffc_get_first_fat_free_cluster0
                             <1>
                             <1> loc_gffc_get_first_fat32_free_cluster:
                             <1>       ; 02/03/2016
0000CFBE E844060000          <1>       call   get_fat32_fsinfo_sector_parms
0000CFC3 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)
0000CFC5 89D0                <1>       mov    eax, edx ; FSI_Next_Free (First Free Cluster)
0000CFC7 83F8FF              <1>       cmp    eax, 0FFFFFFFFh ; invalid (unknown) !
0000CFCA 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:
0000CFCC B802000000          <1>       mov    eax, 2
                             <1>       ;xor   edx, edx
                             <1>
                             <1> loc_gffc_get_first_fat_free_cluster1:
0000CFD1 53                  <1>       push   ebx ; 02/03/2016
                             <1>
                             <1> loc_gffc_get_first_fat_free_cluster2:
0000CFD2 A3[008C0100]        <1>       mov    [gffc_first_free_cluster], eax
0000CFD7 A3[FC8B0100]        <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:
0000CFDC E875FDFFFF              <1>     call    get_next_cluster
0000CFE1 7307                    <1>     jnc     short loc_gffc_get_first_fat_free_cluster4
0000CFE3 09C0                    <1>     or      eax, eax
0000CFE5 740B                    <1>     jz      short loc_gffc_first_free_fat_cluster_next
0000CFE7 5B                      <1>     pop     ebx ; 02/03/2016
0000CFE8 F5                      <1>     cmc     ; stc
0000CFE9 C3                      <1>     retn
                                <1>
                                <1> loc_gffc_get_first_fat_free_cluster4:
0000CFEA 21C0                    <1>     and     eax, eax ; next cluster value
0000CFEC 7504                    <1>     jnz     short loc_gffc_first_free_fat_cluster_next
0000CFEE 89C8                    <1>     mov     eax, ecx ; current (previous cluster) value
0000CFF0 EB22                    <1>     jmp     short loc_gffc_check_for_set
                                <1>
                                <1> loc_gffc_first_free_fat_cluster_next:
0000CFF2 A1[FC8B0100]            <1>     mov     eax, [gffc_next_free_cluster]
0000CFF7 3B05[048C0100]          <1>     cmp     eax, [gffc_last_free_cluster]
0000CFFD 7308                    <1>     jnb     short retn_stc_from_get_first_free_cluster
                                <1> pass_gffc_last_cluster_eax_check:
0000CFFF 40                      <1>     inc     eax ; add eax, 1
0000D000 A3[FC8B0100]            <1>     mov     [gffc_next_free_cluster], eax
0000D005 EBD5                    <1>     jmp     short loc_gffc_get_first_fat_free_cluster3
                                <1>
                                <1> retn_stc_from_get_first_free_cluster:
0000D007 A1[008C0100]            <1>     mov     eax, [gffc_first_free_cluster]
0000D00C 83F802                  <1>     cmp     eax, 2
0000D00F 7709                    <1>     ja      short loc_gffc_check_previous_clusters
0000D011 29C0                    <1>     sub     eax, eax
0000D013 48                      <1>     dec     eax ; FFFFFFFFh
                                <1>
                                <1> loc_gffc_check_for_set:
                                <1>     ; 02/03/2016
0000D014 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>
0000D015 09DB                    <1>     or      ebx, ebx
0000D017 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:
0000D019 C3                      <1>     retn
                                <1>
                                <1> loc_gffc_check_previous_clusters:
0000D01A 48                      <1>     dec     eax ; sub eax, 1
0000D01B A3[048C0100]            <1>     mov     [gffc_last_free_cluster], eax
0000D020 B802000000              <1>     mov     eax, 2
                                <1>     ;xor   edx, edx
0000D025 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'
0000D027 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>
0000D02A 813B52526141            <1>     cmp     dword [ebx], 41615252h
0000D030 7540                    <1>     jne     short loc_sffc_read_fsinfo_sector
0000D032 81BBE4010000727241-     <1>     cmp     dword [ebx+484], 61417272h
0000D03B 61                      <1>
0000D03C 7534                    <1>     jne     short loc_sffc_read_fsinfo_sector
                                <1>
0000D03E 3B83EC010000            <1>     cmp     eax, [ebx+492]  ; FSI_Next_Free
0000D044 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'
```

```
0000D046 8983EC010000        <1>       mov    [ebx+492], eax
0000D04C A1[148C0100]        <1>       mov    eax, [CFS_FAT32FSINFOSEC]
0000D051 B901000000          <1>       mov    ecx, 1
0000D056 53                  <1>       push   ebx
0000D057 E8B2520000          <1>       call   disk_write
0000D05C 7208                <1>       jc      short loc_sffc_read_fsinfo_sector_err1
0000D05E 5B                  <1>       pop    ebx
                             <1>
0000D05F 8B83EC010000        <1>       mov    eax, [ebx+492] ; First (Next) Free Cluster
                             <1>
                             <1> loc_sffc_retn:
0000D065 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:
0000D066 BB00000000          <1>       mov    ebx, 0
                             <1>       ; 15/10/2016 (1Dh -> 18)
                             <1>       ; 23/03/2016 (1Dh)
0000D06B B812000000          <1>       mov    eax, 18 ; Drive not ready or write error
                             <1>
                             <1> loc_sffc_read_fsinfo_sector_err2:
0000D070 5A                  <1>       pop    edx
0000D071 C3                  <1>       retn
                             <1>
                             <1> loc_sffc_read_fsinfo_sector:
0000D072 50                  <1>       push   eax
                             <1>
0000D073 E88F050000          <1>       call   get_fat32_fsinfo_sector_parms
0000D078 72F6                <1>       jc     short loc_sffc_read_fsinfo_sector_err2
                             <1>
0000D07A 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)
0000D07B 39D0                <1>       cmp    eax, edx ; First free Cluster (eax = new value)
0000D07D 75C7                <1>       jne    short loc_sffc_write_fsinfo_sector
                             <1>
0000D07F 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>
0000D080 A3[66890100]        <1>       mov    [FAT_CurrentCluster], eax
0000D085 890D[088C0100]      <1>       mov    [ClusterValue], ecx
                             <1>
                             <1> loc_update_cluster_check_fat_buffer:
0000D08B 8A1E                <1>       mov    bl, [esi+LD_Name]
0000D08D 381D[6B890100]      <1>       cmp    [FAT_BuffDrvName], bl
0000D093 741A                <1>       je     short loc_update_cluster_check_fat_type
0000D095 803D[6A890100]02    <1>       cmp    byte [FAT_BuffValidData], 2
0000D09C 0F84C2000000        <1>         je      loc_uc_save_fat_buffer
                             <1>
                             <1> loc_uc_reset_fat_buffer_validation:
0000D0A2 C605[6A890100]00    <1>       mov    byte [FAT_BuffValidData], 0
                             <1>
                             <1> loc_uc_check_fat_type_reset_drvname:
0000D0A9 881D[6B890100]      <1>       mov    [FAT_BuffDrvName], bl
                             <1>
                             <1> loc_update_cluster_check_fat_type:
0000D0AF 29D2                <1>       sub    edx, edx ; 26/02/2016
0000D0B1 8A5E03              <1>       mov    bl, [esi+LD_FATType]
0000D0B4 83F802              <1>       cmp    eax, 2
0000D0B7 0F82BE000000        <1>         jb      update_cluster_inv_data
0000D0BD 80FB02              <1>       cmp    bl, 2
0000D0C0 0F877A010000        <1>         ja      update_fat32_cluster
                             <1>       ;cmp   bl, 1
                             <1>       ;jb    short update_cluster_inv_data
0000D0C6 8B4E78              <1>       mov    ecx, [esi+LD_Clusters]
0000D0C9 41                  <1>       inc    ecx
0000D0CA 890D[76890100]      <1>       mov    [LastCluster], ecx
0000D0D0 39C8                <1>       cmp    eax, ecx ; dword [LastCluster]
0000D0D2 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:')
0000D0D8 80FB01         <1>       cmp   bl, 1 ; correct comparison is this !
0000D0DB 0F86A2000000   <1>         jna     update_fat12_cluster
                        <1>
                        <1> update_fat16_cluster:
                        <1> pass_uc_fat16_errc:
                        <1>       ;sub   edx, edx
0000D0E1 BB00030000     <1>       mov   ebx, 300h ;768
0000D0E6 F7F3           <1>       div   ebx
                        <1>       ; EAX = Count of 3 FAT sectors
                        <1>       ; DX = Cluster offset in FAT buffer
0000D0E8 6689D3         <1>       mov   bx, dx
0000D0EB 66D1E3         <1>       shl   bx, 1 ; Multiply by 2
0000D0EE 66BA0300       <1>       mov   dx, 3
0000D0F2 F7E2           <1>       mul   edx
                        <1>       ; EAX = FAT Sector
                        <1>       ; EDX = 0
                        <1>       ; EBX = Byte offset in FAT buffer
0000D0F4 8A0D[6A890100] <1>       mov   cl, [FAT_BuffValidData]
0000D0FA 80F902         <1>       cmp   cl, 2
0000D0FD 750A           <1>       jne   short loc_uc_check_fat16_buff_sector_load
                        <1>
                        <1> loc_uc_check_fat16_buff_sector_save:
0000D0FF 3B05[6E890100] <1>       cmp   eax, [FAT_BuffSector]
0000D105 755D           <1>       jne   short loc_uc_save_fat_buffer
0000D107 EB15           <1>       jmp   short loc_update_fat16_cell
                        <1>
                        <1> loc_uc_check_fat16_buff_sector_load:
0000D109 80F901         <1>       cmp   cl, 1 ; byte [FAT_BuffValidData]
0000D10C 0F85FB010000   <1>         jne     loc_uc_load_fat_sectors
0000D112 3B05[6E890100] <1>       cmp   eax, [FAT_BuffSector]
0000D118 0F85EF010000   <1>         jne     loc_uc_load_fat_sectors
                        <1>
                        <1> loc_update_fat16_cell:
                        <1> loc_update_fat16_buffer:
0000D11E 81C3001C0900   <1>       add   ebx, FAT_Buffer ; 26/02/2016
                        <1>       ;movzx eax, word [ebx]
0000D124 668B03         <1>       mov   ax, [ebx]
                        <1>       ; 01/03/2016
0000D127 89C2           <1>       mov   edx, eax ; old value of the cluster
0000D129 A3[66890100]   <1>       mov   [FAT_CurrentCluster], eax
0000D12E 8B0D[088C0100] <1>       mov   ecx, [ClusterValue] ; 32 bits
0000D134 66890B         <1>       mov   [ebx], cx ; 16 bits !
                        <1>
0000D137 C605[6A890100]02 <1>     mov   byte [FAT_BuffValidData], 2
                        <1>
0000D13E 6683F802       <1>       cmp   ax, 2
0000D142 723A           <1>       jb    short return_uc_fat_stc
0000D144 3B05[76890100] <1>       cmp   eax, [LastCluster]
0000D14A 7732           <1>       ja    short return_uc_fat_stc
                        <1>
                        <1> loc_fat_buffer_updated:
                        <1>       ; 01/03/2016
0000D14C F8             <1>       clc
                        <1> loc_fat_buffer_stc_1:
0000D14D 9C             <1>       pushf
0000D14E 21C9           <1>       and   ecx, ecx
0000D150 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
0000D152 FF05[72890100] <1>       inc   dword [FAT_ClusterCounter]
                        <1>
                        <1> loc_fat_buffer_updated_1: ; new value of the cluster > 0
0000D158 09D2           <1>       or    edx, edx ; 02/03/2016
0000D15A 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
0000D15C FF0D[72890100] <1>       dec   dword [FAT_ClusterCounter] ; it may be negative number
                        <1>
                        <1> loc_fat_buffer_updated_2:
0000D162 9D             <1>       popf
0000D163 C3             <1>       retn
                        <1>
                        <1> loc_uc_save_fat_buffer:
                        <1>       ; byte [FAT_BuffValidData] = 2
0000D164 E8D4010000     <1>       call  save_fat_buffer
0000D169 0F8297010000   <1>         jc      loc_fat_sectors_rw_error2
                        <1>       ;mov  byte [FAT_BuffValidData], 1
0000D16F A1[66890100]   <1>       mov   eax, [FAT_CurrentCluster]
                        <1>       ;mov  ecx, [ClusterValue]
                        <1>       ;jmp  short loc_update_cluster_check_fat_buffer
0000D174 8A1E           <1>       mov   bl, [esi+LD_Name] ; 01/03/2016
0000D176 E927FFFFFF     <1>         jmp     loc_uc_reset_fat_buffer_validation
                        <1>
                        <1> update_cluster_inv_data:
                        <1>       ;mov  eax, 0Dh
0000D17B B00D           <1>       mov   al, 0Dh  ; Invalid Data
0000D17D C3             <1>       retn
                        <1>
                        <1> return_uc_fat_stc:
                        <1>       ; 01/03/2016
0000D17E 31C0           <1>       xor   eax, eax
0000D180 F9             <1>       stc
0000D181 EBCA           <1>       jmp   short loc_fat_buffer_stc_1
                        <1>
                        <1> update_fat12_cluster:
                        <1> pass_uc_fat12_errc:
```

```
                                <1>       ;sub   edx, edx
0000D183 BB00040000             <1>       mov    ebx, 400h ;1024
0000D188 F7F3                   <1>       div    ebx
                                <1>       ; EAX = Count of 3 FAT sectors
                                <1>       ; DX = Cluster offset in FAT buffer
0000D18A 66B90300               <1>       mov    cx, 3
0000D18E 6689C3                 <1>       mov    bx, ax
0000D191 6689C8                 <1>       mov    ax, cx ; 3
0000D194 66F7E2                 <1>       mul    dx     ; Multiply by 3
0000D197 66D1E8                 <1>       shr    ax, 1  ; Divide by 2
0000D19A 6693                   <1>       xchg   bx, ax
                                <1>       ; EAX = Count of 3 FAT sectors
                                <1>       ; EBX = Byte Offset in FAT buffer
0000D19C 66F7E1                 <1>       mul    cx  ; 3 * AX
                                <1>       ; EAX = FAT Beginning Sector
                                <1>       ; EDX = 0
0000D19F 8A0D[6A890100]         <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 !)
0000D1A5 80F902                 <1>       cmp    cl, 2 ; 2 = dirty buffer (must be written to disk)
0000D1A8 750A                   <1>       jne    short loc_uc_check_fat12_buff_sector_load
                                <1>
                                <1> loc_uc_check_fat12_buff_sector_save:
0000D1AA 3B05[6E890100]         <1>       cmp    eax, [FAT_BuffSector]
0000D1B0 75B2                   <1>         jne     short loc_uc_save_fat_buffer
0000D1B2 EB15                   <1>       jmp    short loc_update_fat12_cell
                                <1>
                                <1> loc_uc_check_fat12_buff_sector_load:
0000D1B4 80F901                 <1>       cmp    cl, 1 ; byte ptr [FAT_BuffValidData]
0000D1B7 0F8550010000           <1>         jne     loc_uc_load_fat_sectors
0000D1BD 3B05[6E890100]         <1>       cmp    eax, [FAT_BuffSector]
0000D1C3 0F8544010000           <1>         jne     loc_uc_load_fat_sectors
                                <1>
                                <1> loc_update_fat12_cell:
0000D1C9 81C3001C0900           <1>       add    ebx, FAT_Buffer ; 26/02/2016
0000D1CF 668B0D[66890100]       <1>       mov    cx, [FAT_CurrentCluster]
0000D1D6 66D1E9                 <1>       shr    cx, 1
0000D1D9 668B03                 <1>       mov    ax, [ebx]
0000D1DC 6689C2                 <1>       mov    dx, ax
0000D1DF 7344                   <1>       jnc    short uc_fat12_nc_even
                                <1>
0000D1E1 6683E00F               <1>       and    ax, 0Fh
0000D1E5 8B0D[088C0100]         <1>       mov    ecx, [ClusterValue] ; 32 bits
0000D1EB 66C1E104               <1>       shl    cx, 4
0000D1EF 6609C1                 <1>       or     cx, ax
0000D1F2 6689D0                 <1>       mov    ax, dx
0000D1F5 66890B                 <1>       mov    [ebx], cx  ; 16 bits !
0000D1F8 66C1E804               <1>       shr    ax, 4 ; al(bit4..7)+ah(bit0..7)
                                <1>
                                <1> update_fat12_buffer:
0000D1FC A3[66890100]           <1>       mov    [FAT_CurrentCluster], eax
0000D201 89C2                   <1>       mov    edx, eax ; 01/03/2016
0000D203 C605[6A890100]02       <1>       mov    byte [FAT_BuffValidData], 2
0000D20A 6683F802               <1>       cmp    ax, 2
0000D20E 0F826AFFFFFF           <1>         jb      return_uc_fat_stc
0000D214 3B05[76890100]         <1>       cmp    eax, [LastCluster]
0000D21A 0F875EFFFFFF           <1>         ja      return_uc_fat_stc
0000D220 E927FFFFFF             <1>         jmp     loc_fat_buffer_updated
                                <1>
                                <1> uc_fat12_nc_even:
0000D225 662500F0               <1>       and    ax, 0F000h
0000D229 8B0D[088C0100]         <1>       mov    ecx, [ClusterValue] ; 32 bits
0000D22F 80E50F                 <1>       and    ch, 0Fh
0000D232 6609C1                 <1>       or     cx, ax
0000D235 6689D0                 <1>       mov    ax, dx
0000D238 66890B                 <1>       mov    [ebx], cx ; 16 bits !
0000D23B 80E40F                 <1>       and    ah, 0Fh ; al(bit0..7)+ah(bit0..3)
0000D23E EBBC                   <1>       jmp    short update_fat12_buffer
                                <1>
                                <1> update_fat32_cluster:
0000D240 8B4E78                 <1>       mov    ecx, [esi+LD_Clusters]
0000D243 41                     <1>       inc    ecx
0000D244 890D[76890100]         <1>       mov    [LastCluster], ecx
                                <1>
0000D24A 39C8                   <1>       cmp    eax, ecx
0000D24C 0F872CFFFFFF           <1>         ja      return_uc_fat_stc
                                <1>
                                <1> pass_uc_fat32_errc:
                                <1>       ;sub   edx, edx
0000D252 BB80010000             <1>       mov    ebx, 180h ;384
0000D257 F7F3                   <1>       div    ebx
                                <1>       ; EAX = Count of 3 FAT sectors
                                <1>       ; DX = Cluster offset in FAT buffer
0000D259 89D3                   <1>       mov    ebx, edx
0000D25B C1E302                 <1>       shl    ebx, 2 ; Multiply by 4
0000D25E BA03000000             <1>       mov    edx, 3
0000D263 F7E2                   <1>       mul    edx
                                <1>       ; EBX = Cluster Offset in FAT buffer
                                <1>       ; EAX = FAT Sector
                                <1>       ; EDX = 0
0000D265 8A0D[6A890100]         <1>       mov    cl, [FAT_BuffValidData]
0000D26B 80F902                 <1>       cmp    cl, 2
0000D26E 750E                   <1>       jne    short loc_uc_check_fat32_buff_sector_load
                                <1>
                                <1> loc_uc_check_fat32_buff_sector_save:
0000D270 3B05[6E890100]         <1>       cmp    eax, [FAT_BuffSector]
0000D276 0F85E8FEFFFF           <1>         jne     loc_uc_save_fat_buffer
0000D27C EB11                   <1>       jmp    short loc_update_fat32_cell
                                <1>
                                <1> loc_uc_check_fat32_buff_sector_load:
0000D27E 80F901                 <1>       cmp    cl, 1 ; byte [FAT_BuffValidData]
0000D281 0F8586000000           <1>         jne     loc_uc_load_fat_sectors
0000D287 3B05[6E890100]         <1>       cmp    eax, [FAT_BuffSector]
0000D28D 757E                   <1>         jne     loc_uc_load_fat_sectors
```

```
                            <1>
                            <1> loc_update_fat32_cell:
                            <1> loc_update_fat32_buffer:
0000D28F 81C3001C0900       <1>       add   ebx, FAT_Buffer ; 26/02/2016
0000D295 8B03               <1>       mov   eax, [ebx]
0000D297 25FFFFFF0F         <1>       and   eax, 0FFFFFFFh ; 28 bit cluster value
                            <1>
0000D29C 8B15[66890100]     <1>       mov   edx, [FAT_CurrentCluster] ; 01/03/2016
                            <1>
0000D2A2 A3[66890100]       <1>       mov   [FAT_CurrentCluster], eax
0000D2A7 8B0D[088C0100]     <1>       mov   ecx, [ClusterValue]
0000D2AD 890B               <1>       mov   [ebx], ecx ; 29/02/2016
                            <1>
0000D2AF C605[6A890100]02   <1>       mov   byte [FAT_BuffValidData], 2
                            <1>
                            <1>       ; 01/03/2016
0000D2B6 21C0               <1>       and   eax, eax ; was it free cluster ?
0000D2B8 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>
0000D2BA 3B563E             <1>       cmp   edx, [esi+LD_BPB+BPB_Reserved+4] ; First free cluster
0000D2BD 7520               <1>       jne   short loc_upd_fat32_c3
                            <1>
0000D2BF 3B15[76890100]     <1>       cmp   edx, [LastCluster]
0000D2C5 7207               <1>       jb    short loc_upd_fat32_c0
                            <1>
0000D2C7 BA02000000         <1>       mov   edx, 2 ; rewind !
0000D2CC EB0E               <1>       jmp   short loc_upd_fat32_c2
                            <1>
                            <1> loc_upd_fat32_c0:
0000D2CE FF463E             <1>       inc   dword [esi+LD_BPB+BPB_Reserved+4] ; set it to next cluster
0000D2D1 EB0C               <1>       jmp   short loc_upd_fat32_c3
                            <1>
                            <1> loc_upd_fat32_c1:
0000D2D3 09C9               <1>       or    ecx, ecx ; will it be free cluster ?
0000D2D5 7508               <1>       jnz   short loc_upd_fat32_c3
                            <1>
0000D2D7 3B563E             <1>       cmp   edx, [esi+LD_BPB+BPB_Reserved+4] ; First free cluster
0000D2DA 7303               <1>       jnb   short loc_upd_fat32_c3
                            <1>
                            <1> loc_upd_fat32_c2:
0000D2DC 89563E             <1>       mov   [esi+LD_BPB+BPB_Reserved+4], edx
                            <1>
                            <1> loc_upd_fat32_c3:
0000D2DF 89C2               <1>       mov   edx, eax
                            <1>
                            <1> loc_upd_fat32_c4:
0000D2E1 83F802             <1>       cmp   eax, 2
0000D2E4 0F8294FEFFFF       <1>         jb      return_uc_fat_stc
                            <1>
                            <1> pass_uc_fat32_c_zero_check_2:
0000D2EA 3B05[76890100]     <1>       cmp   eax, [LastCluster]
0000D2F0 0F8788FEFFFF       <1>         ja      return_uc_fat_stc
                            <1>
0000D2F6 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
0000D2FB B811000000         <1>       mov   eax, 17 ; Drive not ready or read error
0000D300 8825[6A890100]     <1>       mov   [FAT_BuffValidData], ah ; 0
                            <1>
                            <1> loc_fat_sectors_rw_error2:
                            <1>       ;mov  eax, error code
                            <1>       ;mov  edx, 0
0000D306 8B0D[088C0100]     <1>       mov   ecx, [ClusterValue]
0000D30C C3                 <1>       retn
                            <1>
                            <1> loc_uc_load_fat_sectors:
0000D30D A3[6E890100]       <1>       mov   [FAT_BuffSector], eax
                            <1>
                            <1> load_uc_fat_sectors_zero:
0000D312 034660             <1>       add   eax, [esi+LD_FATBegin]
0000D315 BB001C0900         <1>       mov   ebx, FAT_Buffer
0000D31A B903000000         <1>       mov   ecx, 3
0000D31F E8F94F0000         <1>       call  disk_read
0000D324 72D5               <1>       jc    short loc_fat_sectors_rw_error1
                            <1>
0000D326 C605[6A890100]01   <1>         mov     byte [FAT_BuffValidData], 1
0000D32D A1[66890100]       <1>       mov   eax, [FAT_CurrentCluster]
0000D332 8B0D[088C0100]     <1>       mov   ecx, [ClusterValue]
0000D338 E972FDFFFF         <1>         jmp     loc_update_cluster_check_fat_type
                            <1>
                            <1> save_fat_buffer:
                            <1>       ; 15/10/2016
                            <1>       ; 01/03/2016
                            <1>       ; 22/02/2016 (TRDOS 386 =  TRDOS v2.0)
                            <1>       ; 11/08/2011
                            <1>       ; 09/02/2005
                            <1>       ; INPUT ->
                            <1>       ;     None
                            <1>       ; OUTPUT ->
                            <1>       ;     cf = 0 -> OK.
                            <1>       ;     cf = 1 -> error code in AL (EAX)
                            <1>       ;
                            <1>       ;     EBX = FAT_Buffer address
                            <1>       ;
                            <1>       ; (EAX, EDX, ECX will be modified)
                            <1>
                            <1>       ;cmp  byte [FAT_BuffValidData], 2
                            <1>       ;je   short loc_save_fat_buff
                            <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 0000D33D 31D2                   <1>         xor     edx, edx
1012 0000D33F 8A35[6B890100]         <1>         mov     dh, [FAT_BuffDrvName]
1013 0000D345 80FE41                 <1>         cmp     dh, 'A'
1014 0000D348 722E                   <1>         jb      short loc_save_fat_buffer_inv_data_retn
1015 0000D34A 80EE41                 <1>         sub     dh, 'A'
1016 0000D34D 56                     <1>         push    esi ; *
1017 0000D34E BE00010900             <1>           mov     esi, Logical_DOSDisks
1018 0000D353 01D6                   <1>         add     esi, edx
1019                                 <1>
1020 0000D355 8A5603                 <1>         mov     dl, [esi+LD_FATType]
1021 0000D358 20D2                   <1>         and     dl, dl
1022 0000D35A 741B                   <1>         jz      short loc_save_fat_buffer_inv_data_pop_retn
1023                                 <1>
1024 0000D35C A1[6E890100]           <1>         mov     eax, [FAT_BuffSector]
1025 0000D361 80FA02                 <1>         cmp     dl, 2
1026 0000D364 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 0000D366 0FB74E1C               <1>         movzx   ecx, word [esi+LD_BPB+FATSecs]
1035 0000D36A 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 0000D36C 7609                   <1>         jna     short loc_save_fat_buffer_inv_data_pop_retn ; correct addr.
1039 0000D36E EB15                   <1>         jmp     short loc_save_fat_buffer_check_rs3
1040                                 <1>
1041                                 <1> loc_save_fat32_buff:
1042 0000D370 8B4E2A                 <1>         mov     ecx, [esi+LD_BPB+FAT32_FAT_Size]
1043 0000D373 29C1                   <1>         sub     ecx, eax
1044 0000D375 770E                   <1>         ja      short loc_save_fat_buffer_check_rs3
1045                                 <1>
1046                                 <1> loc_save_fat_buffer_inv_data_pop_retn:
1047 0000D377 5E                     <1>         pop     esi ; *
1048                                 <1> loc_save_fat_buffer_inv_data_retn:
1049 0000D378 B80D000000             <1>         mov     eax, 0Dh ; Invalid DATA
1050 0000D37D C3                     <1>         retn
1051                                 <1>
1052                                 <1> loc_save_fat_buff_remain_sectors_3:
1053 0000D37E B903000000             <1>         mov     ecx, 3
1054 0000D383 EB05                   <1>         jmp     short loc_save_fat_buff_continue
1055                                 <1>
1056                                 <1> loc_save_fat_buffer_check_rs3:
1057 0000D385 83F903                 <1>         cmp     ecx, 3
1058 0000D388 77F4                   <1>         ja      short loc_save_fat_buff_remain_sectors_3
1059                                 <1>
1060                                 <1> loc_save_fat_buff_continue:
1061 0000D38A BB001C0900             <1>         mov     ebx, FAT_Buffer
1062 0000D38F 034660                 <1>         add     eax, [esi+LD_FATBegin]
1063 0000D392 51                     <1>         push    ecx
1064 0000D393 E8764F0000             <1>         call    disk_write
1065 0000D398 59                     <1>         pop     ecx
1066 0000D399 722B                   <1>         jc      short loc_save_FAT_buff_write_err
1067                                 <1>
1068 0000D39B 807E0302               <1>         cmp     byte [esi+LD_FATType], 2
1069 0000D39F 7605                   <1>         jna     short loc_calc_2nd_fat12_16_addr
1070                                 <1>
1071                                 <1> loc_calc_2nd_fat32_addr:
1072 0000D3A1 8B462A                 <1>         mov     eax, [esi+LD_BPB+FAT32_FAT_Size]
1073 0000D3A4 EB04                   <1>         jmp     short loc_calc_2nd_fat_addr
1074                                 <1>
1075                                 <1> loc_calc_2nd_fat12_16_addr:
1076 0000D3A6 0FB7461C               <1>         movzx   eax, word [esi+LD_BPB+FATSecs]
1077                                 <1>
1078                                 <1> loc_calc_2nd_fat_addr:
1079 0000D3AA 034660                 <1>         add     eax, [esi+LD_FATBegin]
1080 0000D3AD 0305[6E890100]         <1>         add     eax, [FAT_BuffSector]
1081 0000D3B3 BB001C0900             <1>         mov     ebx, FAT_Buffer
1082                                 <1>         ; ecx = 1 to 3
1083 0000D3B8 E8514F0000             <1>         call    disk_write
1084 0000D3BD 7207                   <1>         jc      short loc_save_FAT_buff_write_err
1085                                 <1>         ; Valid  buffer (1 = valid but do not save)
1086 0000D3BF C605[6A890100]01       <1>         mov     byte [FAT_BuffValidData], 1
1087                                 <1>
1088                                 <1> loc_save_FAT_buff_write_err:
1089 0000D3C6 5E                     <1>         pop     esi ; *
1090 0000D3C7 BB001C0900             <1>         mov     ebx, FAT_Buffer
1091                                 <1>         ; 15/10/2016 (1Dh -> 18)
1092                                 <1>         ; 23/03/2016 (1Dh)
1093 0000D3CC B812000000             <1>         mov     eax, 18 ; Drive not ready or write error
1094 0000D3D1 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)
```

```
                                <1>       ; OUTPUT ->
                                <1>       ;      EAX = Free Space in sectors
                                <1>       ;      ESI = Logical Dos Drive Description Table address
                                <1>       ;      BH = Logical Dos Drive Number (same with input value of BH)
                                <1>       ;      BL = Type of operation (same with input value of BL)
                                <1>       ;      ECX = 0 -> valid
                                <1>       ;      ECX > 0 -> error or invalid
                                <1>       ;      If EAX = FFFFFFFFh, it is 're-calculation needed'
                                <1>       ;                          sign due to r/w error
                                <1>
0000D3D2 66891D[0E8C0100]       <1>       mov    [CFS_OPType], bx
0000D3D9 A3[108C0100]           <1>       mov    [CFS_CC], eax
                                <1>
0000D3DE 80FFFF                 <1>       cmp    bh, 0FFh
0000D3E1 740B                   <1>       je     short pass_calculate_freespace_get_drive_dt_offset
                                <1>
                                <1> loc_calculate_freespace_get_drive_dt_offset:
0000D3E3 31C0                   <1>       xor    eax, eax
0000D3E5 88FC                   <1>         mov     ah, bh
0000D3E7 BE00010900             <1>       mov    esi, Logical_DOSDisks
0000D3EC 01C6                   <1>         add     esi, eax
                                <1>
                                <1> pass_calculate_freespace_get_drive_dt_offset:
0000D3EE 08DB                   <1>       or     bl, bl
0000D3F0 7435                   <1>       jz     short loc_reset_fcc
                                <1>
                                <1> loc_get_free_sectors:
0000D3F2 8B4674                 <1>       mov    eax, [esi+LD_FreeSectors]
                                <1>
                                <1>       ;xor   ecx, ecx
                                <1>       ;dec   ecx ; 0FFFFFFFFh
                                <1>       ;cmp   eax, ecx ; 29/02/2016
                                <1>       ;je    short loc_get_free_sectors_retn ; recalculation is needed!
                                <1>
                                <1>       ; 23/03/2016
0000D3F5 8B4E70                 <1>       mov    ecx, [esi+LD_TotalSectors]
0000D3F8 39C1                   <1>       cmp    ecx, eax ; Total sectors must be greater than Free sectors !
0000D3FA 7707                   <1>       ja     short loc_get_free_sectors_check_optype
                                <1>
0000D3FC 31C0                   <1>       xor    eax, eax
0000D3FE 48                     <1>       dec    eax ; 0FFFFFFFFh  ; recalculation is needed!
0000D3FF 894674                 <1>       mov    [esi+LD_FreeSectors], eax ; reset (for recalculation)
                                <1>
                                <1> loc_get_free_sectors_retn:
0000D402 C3                     <1>       retn
                                <1>
                                <1> loc_get_free_sectors_check_optype:
0000D403 80FB03                 <1>       cmp    bl, 3
0000D406 7203                   <1>       jb     short loc_set_fcc
                                <1>
0000D408 29C9                   <1>       sub    ecx, ecx ; 0
                                <1>
0000D40A C3                     <1>       retn
                                <1>
                                <1> loc_set_fcc:
0000D40B 807E0302               <1>       cmp    byte [esi+LD_FATType], 2
0000D40F 0F87DF000000           <1>         ja      loc_update_FAT32_fs_info_fcc
                                <1>
                                <1>       ;mov   eax, [esi+LD_FreeSectors]
0000D415 0FB64E13               <1>       movzx  ecx, byte [esi+LD_BPB+SecPerClust]
0000D419 29D2                   <1>       sub    edx, edx
0000D41B F7F1                   <1>       div    ecx
                                <1>       ;or    dx, dx
                                <1>       ;      ; DX -> Remain sectors < SecPerClust
                                <1>       ;      ; DX > 0 -> invalid free sector count
                                <1>       ;jnz   short loc_reset_fcc
                                <1>
                                <1> ;pass_set_fcc_div32:
0000D41D A3[87890100]           <1>       mov    [FreeClusterCount], eax
0000D422 E988000000             <1>         jmp     loc_set_free_sectors_FAT12_FAT16
                                <1>
                                <1> loc_reset_fcc:
0000D427 31C0                   <1>       xor    eax, eax
0000D429 A3[87890100]           <1>       mov    [FreeClusterCount], eax ; 0
0000D42E 8B5678                 <1>       mov    edx, [esi+LD_Clusters]
0000D431 42                     <1>       inc    edx
0000D432 8915[76890100]         <1>       mov    [LastCluster], edx
                                <1>
0000D438 807E0302               <1>       cmp    byte [esi+LD_FATType], 2
0000D43C 7647                   <1>       jna    short loc_count_free_fat_clusters_0
                                <1>
0000D43E 48                     <1>       dec    eax ; FFFFFFFFh
0000D43F A3[188C0100]           <1>       mov    [CFS_FAT32FC], eax
                                <1>
                                <1>       ; 29/02/2016
0000D444 89463A                 <1>       mov    [esi+LD_BPB+BPB_Reserved], eax ; reset
0000D447 89463E                 <1>       mov    [esi+LD_BPB+BPB_Reserved+4], eax ; reset
                                <1>
0000D44A B802000000             <1>       mov    eax, 2
                                <1>
                                <1> loc_count_fc_next_cluster_0:
0000D44F 50                     <1>       push   eax
0000D450 E801F9FFFF             <1>       call   get_next_cluster
0000D455 7310                   <1>       jnc    short loc_check_fat32_ff_cluster
0000D457 09C0                   <1>       or     eax, eax
0000D459 741E                   <1>       jz     short pass_inc_cfs_fcc_0
                                <1>
                                <1> loc_put_fcc_unknown_sign:
0000D45B 58                     <1>       pop    eax
                                <1>       ; "Free count is Unknown" sign
                                <1>       ;mov   dword [FreeClusterCount], 0FFFFFFFFh
                                <1>
                                <1>       ; 29/02/2016
                                <1>       ; Save Free Cluster Count value in FAT32 'BPB_Reserved' area
                                <1>       ;mov   [esi+LD_BPB+BPB_Reserved], 0FFFFFFFFh ; unknown!
```

```
1216 0000D45C 8B15[188C0100]      <1>       mov    edx, [CFS_FAT32FC] ; First Free Cluster
1217                              <1>       ; Save First Free Cluster value in FAT32 'BPB_Reserved+4' area
1218 0000D462 89563E              <1>       mov    [esi+LD_BPB+BPB_Reserved+4], edx
1219                              <1>
1220 0000D465 EB7D                <1>         jmp     loc_put_fcc_invalid_sign
1221                              <1>
1222                              <1> loc_check_fat32_ff_cluster:
1223 0000D467 09C0                <1>       or     eax, eax
1224 0000D469 750E                <1>       jnz    short pass_inc_cfs_fcc_0
1225 0000D46B 58                  <1>       pop    eax
1226 0000D46C A3[188C0100]        <1>       mov    [CFS_FAT32FC], eax
1227                              <1>       ;mov   dword [FreeClusterCount], 1
1228 0000D471 FF05[87890100]      <1>       inc    dword [FreeClusterCount]
1229 0000D477 EB27                <1>       jmp    short pass_inc_cfs_fcc_1
1230                              <1>
1231                              <1> pass_inc_cfs_fcc_0:
1232 0000D479 58                  <1>       pop    eax
1233                              <1>
1234                              <1> pass_inc_cfs_fcc_0c:
1235 0000D47A 40                  <1>       inc    eax ; add eax, 1
1236 0000D47B 3B05[76890100]      <1>       cmp    eax, [LastCluster]
1237 0000D481 76CC                <1>       jna    short loc_count_fc_next_cluster_0
1238 0000D483 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 0000D485 B002                <1>       mov    al, 2
1243                              <1>
1244                              <1> loc_count_fc_next_cluster:
1245 0000D487 50                  <1>       push   eax
1246 0000D488 E8C9F8FFFF          <1>       call   get_next_cluster
1247 0000D48D 720C                <1>       jc     short loc_count_fcc_stc
1248                              <1>
1249                              <1> loc_count_free_clusters_1:
1250 0000D48F 21C0                <1>       and    eax, eax
1251 0000D491 750C                <1>       jnz    short pass_inc_cfs_fcc
1252                              <1>
1253 0000D493 FF05[87890100]      <1>       inc    dword [FreeClusterCount]
1254 0000D499 EB04                <1>       jmp    short pass_inc_cfs_fcc
1255                              <1>
1256                              <1> loc_count_fcc_stc:
1257 0000D49B 09C0                <1>       or     eax, eax
1258 0000D49D 75BC                <1>       jnz    short loc_put_fcc_unknown_sign ; 29/02/2016
1259                              <1>
1260                              <1> pass_inc_cfs_fcc:
1261 0000D49F 58                  <1>       pop    eax
1262                              <1>
1263                              <1> pass_inc_cfs_fcc_1:
1264 0000D4A0 40                  <1>       inc    eax ; add eax, 1
1265 0000D4A1 3B05[76890100]      <1>       cmp    eax, [LastCluster]
1266 0000D4A7 76DE                <1>       jna    short loc_count_fc_next_cluster
1267                              <1>
1268                              <1> loc_set_free_sectors:
1269 0000D4A9 807E0302            <1>       cmp    byte [esi+LD_FATType], 2
1270 0000D4AD 7745                <1>       ja     short loc_update_FAT32_fs_info_fcc
1271                              <1>
1272                              <1> loc_set_free_sectors_FAT12_FAT16:
1273 0000D4AF 803D[0E8C0100]00    <1>       cmp    byte [CFS_OPType], 0
1274 0000D4B6 761C                <1>       jna    short pass_FAT_add_sub_fcc
1275 0000D4B8 A1[108C0100]        <1>       mov    eax, [CFS_CC]
1276 0000D4BD 803D[0E8C0100]01    <1>       cmp    byte [CFS_OPType], 1
1277 0000D4C4 7708                <1>       ja     short pass_FAT_add_fcc
1278 0000D4C6 0105[87890100]      <1>       add    [FreeClusterCount], eax
1279 0000D4CC EB06                <1>       jmp    short pass_FAT_add_sub_fcc
1280                              <1>
1281                              <1> pass_FAT_add_fcc:
1282 0000D4CE 2905[87890100]      <1>       sub    [FreeClusterCount], eax
1283                              <1>
1284                              <1> pass_FAT_add_sub_fcc:
1285 0000D4D4 0FB64613            <1>       movzx  eax, byte [esi+LD_BPB+SecPerClust]
1286 0000D4D8 8B15[87890100]      <1>       mov    edx, [FreeClusterCount]
1287 0000D4DE F7E2                <1>       mul    edx
1288                              <1>
1289 0000D4E0 31C9                <1>       xor    ecx, ecx
1290 0000D4E2 EB05                <1>       jmp    short loc_cfs_retn_params
1291                              <1>
1292                              <1> loc_put_fcc_invalid_sign:
1293 0000D4E4 29C0                <1>             sub    eax, eax ; 0
1294 0000D4E6 48                  <1>       dec    eax ; FFFFFFFFh
1295                              <1> loc_fat32_ffc_recalc_needed:
1296 0000D4E7 89C1                <1>       mov    ecx, eax
1297                              <1>
1298                              <1> loc_cfs_retn_params:
1299 0000D4E9 894674              <1>       mov    [esi+LD_FreeSectors], eax
1300 0000D4EC 0FB71D[0E8C0100]    <1>       movzx  ebx, word [CFS_OPType]
1301 0000D4F3 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 0000D4F4 803D[0E8C0100]01    <1>       cmp    byte [CFS_OPType], 1
1309 0000D4FB 7221                <1>       jb     short loc_cfs_FAT32_get_rcalc_parms ; 0 = recalculated
1310 0000D4FD 7406                <1>       je     short loc_check_fcc_FSINFO_op1 ; 1 = add
1311                              <1> loc_check_fcc_FSINFO_op2: ; subtract
1312 0000D4FF F71D[108C0100]      <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 0000D505 31D2                <1>       xor    edx, edx ; 0
1316 0000D507 4A                  <1>       dec    edx ; 0FFFFFFFFh
1317 0000D508 8B463A              <1>       mov    eax, [esi+LD_BPB+BPB_Reserved]
1318 0000D50B 39D0                <1>       cmp    eax, edx
1319 0000D50D 73D5                <1>       jnb    short loc_put_fcc_invalid_sign
1320 0000D50F 0305[108C0100]      <1>         add     eax, [CFS_CC] ; free cluster count on disk + current count
```

```
1321 0000D515 72CD              <1>       jc    short loc_put_fcc_invalid_sign
1322                            <1>
1323 0000D517 A3[87890100]      <1>       mov   [FreeClusterCount], eax
1324 0000D51C EB0E              <1>       jmp   short loc_cfs_write_FSINFO_sector
1325                            <1>
1326                            <1> loc_cfs_FAT32_get_rcalc_parms:
1327 0000D51E 8B15[188C0100]    <1>       mov   edx, [CFS_FAT32FC]
1328 0000D524 A1[87890100]      <1>       mov   eax, [FreeClusterCount]
1329 0000D529 89563E            <1>       mov   [esi+LD_BPB+BPB_Reserved+4], edx ; First Free Cluster
1330                            <1> loc_cfs_write_FSINFO_sector:
1331 0000D52C 89463A            <1>       mov   [esi+LD_BPB+BPB_Reserved], eax ; Free cluster count
1332                            <1>       ; 01/03/2016
1333 0000D52F E8AA000000        <1>       call  set_fat32_fsinfo_sector_parms
1334 0000D534 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 0000D536 8B0D[87890100]    <1>       mov   ecx, [FreeClusterCount]
1344 0000D53C 0FB64613          <1>       movzx  eax, byte [esi+LD_BPB+SecPerClust]
1345 0000D540 F7E1              <1>       mul   ecx
1346                            <1>       ; 29/02/2016
1347 0000D542 31C9              <1>       xor   ecx, ecx ; 0
1348 0000D544 09D2              <1>       or    edx, edx ; 0 ?
1349 0000D546 759C              <1>         jnz     loc_put_fcc_invalid_sign
1350 0000D548 394670            <1>       cmp   [esi+LD_TotalSectors], eax ; Volume size in sectors
1351 0000D54B 7697              <1>         jna     short loc_put_fcc_invalid_sign
1352                            <1>       ;
1353                            <1> loc_set_FAT32_free_sectors_ok:
1354 0000D54D 31D2              <1>       xor   edx, edx ; 0
1355 0000D54F 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 0000D551 89C1              <1>       mov   ecx, eax
1377                            <1>
1378 0000D553 C705[208C0100]FFFF- <1>     mov   dword [glc_index], 0FFFFFFFFh ; 22/10/2016
1378 0000D55B FFFF              <1>
1379                            <1>
1380                            <1> loc_glc_get_next_cluster_1:
1381 0000D55D 890D[1C8C0100]    <1>       mov   [glc_prevcluster], ecx
1382                            <1>       ; 22/10/2016
1383 0000D563 FF05[208C0100]    <1>       inc   dword [glc_index]
1384                            <1>
1385                            <1> loc_glc_get_next_cluster_2:
1386 0000D569 E8E8F7FFFF        <1>       call  get_next_cluster
1387                            <1>       ; ecx = current/previous cluster
1388                            <1>       ; eax = next/last cluster
1389 0000D56E 73ED              <1>       jnc   short loc_glc_get_next_cluster_1
1390                            <1>
1391 0000D570 09C0              <1>       or    eax, eax
1392 0000D572 7509              <1>       jnz   short loc_glc_stc_retn
1393                            <1>
1394                            <1>       ; ecx = previous cluster
1395 0000D574 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 0000D576 8B0D[1C8C0100]    <1>       mov   ecx, [glc_prevcluster]
1402 0000D57C C3                <1>       retn
1403                            <1>
1404                            <1> loc_glc_stc_retn:
1405 0000D57D F5                <1>       cmc   ;stc
1406 0000D57E 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 0000D580 31C9                <1>       xor    ecx, ecx ; mov ecx, 0
1426                              <1>       ;mov   byte [FAT_BuffValidData], 0
1427 0000D582 890D[72890100]      <1>       mov    [FAT_ClusterCounter], ecx ; 0 ; reset
1428                              <1>
1429                              <1> loc_tcc_unlink_clusters:
1430 0000D588 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 0000D58D 73F9                <1>       jnc short loc_tcc_unlink_clusters
1437                              <1>
1438                              <1> pass_tcc_unlink_clusters:
1439 0000D58F A2[278C0100]        <1>       mov    byte [TCC_FATErr], al
1440 0000D594 803D[6A890100]02    <1>       cmp    byte [FAT_BuffValidData], 2
1441 0000D59B 750E                <1>       jne    short loc_tcc_calculate_FAT_freespace
1442 0000D59D E89BFDFFFF          <1>       call   save_fat_buffer
1443 0000D5A2 7307                <1>       jnc    short loc_tcc_calculate_FAT_freespace
1444 0000D5A4 A2[278C0100]        <1>       mov    byte [TCC_FATErr], al ; Error
1445                              <1>       ;mov   byte [FAT_BuffValidData], 0
1446                              <1>
1447                              <1>       ; 01/03/2016
1448 0000D5A9 EB12                <1>       jmp    short loc_tcc_recalculate_FAT_freespace
1449                              <1>
1450                              <1> loc_tcc_calculate_FAT_freespace:
1451 0000D5AB A1[72890100]        <1>       mov    eax, [FAT_ClusterCounter] ; signed (+-) number
1452 0000D5B0 66BB01FF            <1>       mov    bx, 0FF01h ; BH = FFh -> ESI = Dos drv desc. table
1453                              <1>                         ; BL = 1 -> add cluster
1454 0000D5B4 E819FEFFFF          <1>       call   calculate_fat_freespace
1455 0000D5B9 21C9                <1>       and    ecx, ecx ; cx = 0 -> valid free sector count
1456 0000D5BB 7409                <1>       jz     short pass_truncate_cc_recalc_FAT_freespace
1457                              <1>
1458                              <1> loc_tcc_recalculate_FAT_freespace:
1459 0000D5BD 66BB00FF            <1>       mov    bx, 0FF00h ; recalculate !
1460 0000D5C1 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 0000D5C6 8B0D[72890100]      <1>       mov    ecx, [FAT_ClusterCounter]
1465                              <1>
1466 0000D5CC 803D[278C0100]00    <1>       cmp    byte [TCC_FATErr], 0
1467 0000D5D3 7608                <1>       jna    short loc_tcc_unlink_clusters_retn
1468                              <1>
1469                              <1> loc_tcc_unlink_clusters_error:
1470 0000D5D5 0FB605[278C0100]    <1>       movzx  eax, byte [TCC_FATErr]
1471 0000D5DC F9                  <1>       stc
1472                              <1> loc_tcc_unlink_clusters_retn:
1473 0000D5DD 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 0000D5DE E824000000          <1>       call   get_fat32_fsinfo_sector_parms
1491 0000D5E3 7221                <1>       jc     short update_fat32_fsinfo_sector_retn
1492                              <1>
1493 0000D5E5 8B463A              <1>       mov    eax, [esi+LD_BPB+BPB_Reserved] ; Free Cluster Count
1494 0000D5E8 8B563E              <1>       mov    edx, [esi+LD_BPB+BPB_Reserved+4] ; First free Cluster
1495                              <1>
1496                              <1>         ;mov ebx, DOSBootSectorBuff
1497 0000D5EB 8983E8010000        <1>       mov    [ebx+488], eax
1498 0000D5F1 8993EC010000        <1>       mov    [ebx+492], edx
1499                              <1>
1500 0000D5F7 A1[148C0100]        <1>       mov    eax, [CFS_FAT32FSINFOSEC]
1501 0000D5FC B901000000          <1>       mov    ecx, 1
1502 0000D601 E8084D0000          <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 0000D606 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 0000D607 0FB74636                 <1>       movzx  eax, word [esi+LD_BPB+FAT32_FSInfoSec]
1534 0000D60B 03466C                   <1>       add    eax, [esi+LD_StartSector]
1535 0000D60E A3[148C0100]             <1>       mov    [CFS_FAT32FSINFOSEC], eax
1536                                    <1>
1537 0000D613 BB[66870100]             <1>         mov     ebx, DOSBootSectorBuff
1538 0000D618 B901000000               <1>       mov    ecx, 1
1539 0000D61D E8FB4C0000               <1>       call   disk_read
1540 0000D622 7232                     <1>       jc     short loc_read_FAT32_fsinfo_sec_err
1541                                    <1>
1542 0000D624 BB[66870100]             <1>       mov    ebx, DOSBootSectorBuff
1543                                    <1>
1544 0000D629 813B52526141             <1>       cmp    dword [ebx], 41615252h
1545 0000D62F 751E                     <1>       jne    short loc_read_FAT32_fsinfo_sec_stc
1546                                    <1>
1547 0000D631 81BBE4010000727241-      <1>       cmp    dword [ebx+484], 61417272h
1547 0000D63A 61                       <1>
1548 0000D63B 7512                     <1>       jne    short loc_read_FAT32_fsinfo_sec_stc
1549                                    <1>
1550 0000D63D A1[148C0100]             <1>       mov    eax, [CFS_FAT32FSINFOSEC]
1551 0000D642 8B8BE8010000             <1>       mov    ecx, [ebx+488] ; free cluster count
1552 0000D648 8B93EC010000             <1>       mov    edx, [ebx+492] ; first (next) free cluster
1553                                    <1>
1554 0000D64E C3                       <1>       retn
1555                                    <1>
1556                                    <1> loc_read_FAT32_fsinfo_sec_stc:
1557                                    <1>       ; 15/10/2016 (0Bh -> 28)
1558 0000D64F B81C000000               <1>       mov    eax, 28 ; Invalid format!
1559 0000D654 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 0000D656 B811000000               <1>       mov    eax, 17 ; Drive not ready or read error
1565                                    <1>
1566                                    <1> loc_read_FAT32_fsinfo_sec_stc_retn:
1567 0000D65B 29DB                     <1>       sub    ebx, ebx ; 0
1568 0000D65D F9                       <1>       stc
1569 0000D65E 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 0000D65F A3[448D0100]             <1>       mov    [FAT_anc_LCluster], eax
1594                                    <1>
1595 0000D664 E844F9FFFF               <1>       call   get_first_free_cluster
1596 0000D669 720B                     <1>       jc     short loc_add_new_cluster_retn
1597                                    <1>       ; EAX >= 2 and EAX < FFFFFFFFh is valid
1598                                    <1>
1599 0000D66B 89C2                     <1>       mov    edx, eax
1600                                    <1>
1601 0000D66D 42                       <1>       inc    edx
1602                                    <1>       ;jnz   short loc_add_new_cluster_check_ffc_eax
1603 0000D66E 7516                     <1>       jnz    short loc_add_new_cluster_save_fcc
1604                                    <1>
1605                                    <1> loc_add_new_cluster_no_disk_space_retn:
1606 0000D670 B827000000               <1>       mov    eax, 27h ; MSDOS err => insufficient disk space
1607                                    <1> loc_add_new_cluster_stc_retn:
1608 0000D675 F9                       <1>       stc
1609                                    <1> loc_add_new_cluster_retn:
1610 0000D676 0FB65E13                 <1>       movzx  ebx, byte [esi+LD_BPB+SecPerClust]
1611 0000D67A 8B4E74                   <1>       mov    ecx, [esi+LD_FreeSectors]
1612                                    <1>       ;xor   edx, edx
1613                                    <1>       ;stc
1614 0000D67D C3                       <1>       retn
1615                                    <1>
1616                                    <1> loc_anc_invalid_format_stc_retn:
1617 0000D67E F9                       <1>       stc
1618                                    <1> loc_add_new_cluster_invalid_format_retn:
1619                                    <1>       ; 15/10/2016 (0Bh -> 28)
1620 0000D67F B81C000000               <1>       mov    eax, 28 ; Invalid format
1621 0000D684 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 0000D686 A3[488D0100]             <1>       mov    [FAT_anc_FFCluster], eax
1629                                    <1>
1630 0000D68B 83E802                   <1>       sub    eax, 2
1631 0000D68E 0FB65E13                 <1>         movzx   ebx, byte [esi+LD_BPB+SecPerClust]
1632 0000D692 F7E3                     <1>       mul    ebx
1633 0000D694 09D2                     <1>       or     edx, edx
```

```
1634 0000D696 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 0000D698 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 0000D699 BF00000700          <1>       mov    edi, Cluster_Buffer ; 70000h (for current TRDOS 386 version)
1651 0000D69E 89D9              <1>       mov    ecx, ebx ; sector count
1652 0000D6A0 C1E107            <1>       shl    ecx, 7 ; 1 sector = 512 bytes -> 128 double words
1653                            <1>       ;xor   eax, eax ; 0
1654 0000D6A3 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 0000D6A5 89D0              <1>       mov    eax, edx
1660 0000D6A7 034668            <1>         add     eax, [esi+LD_DATABegin]
1661 0000D6AA 72D3              <1>       jc     short loc_add_new_cluster_invalid_format_retn
1662                            <1>
1663 0000D6AC 89D9              <1>       mov    ecx, ebx ; ECX = sectors per cluster (<256)
1664 0000D6AE BB00000700        <1>       mov    ebx, Cluster_Buffer
1665 0000D6B3 E8564C0000        <1>       call   disk_write
1666 0000D6B8 7307              <1>       jnc    short loc_add_new_cluster_update_fat_nlc
1667                            <1>
1668                            <1>       ; 15/10/2016 (1Dh -> 18)
1669 0000D6BA B812000000        <1>       mov    eax, 18 ; Write Error
1670 0000D6BF EBB4              <1>       jmp    short loc_add_new_cluster_stc_retn
1671                            <1>
1672                            <1> loc_add_new_cluster_update_fat_nlc:
1673 0000D6C1 A1[488D0100]      <1>       mov    eax, [FAT_anc_FFCluster]
1674 0000D6C6 31C9              <1>       xor    ecx, ecx
1675 0000D6C8 890D[72890100]    <1>       mov    [FAT_ClusterCounter], ecx ; 0 ; reset
1676 0000D6CE 49                <1>       dec    ecx ; 0FFFFFFFFh
1677 0000D6CF E8ACF9FFFF        <1>       call   update_cluster
1678 0000D6D4 7304              <1>       jnc    short loc_add_new_cluster_update_fat_plc
1679 0000D6D6 09C0              <1>       or     eax, eax ;EAX = 0 -> cluster value is 0 or eocc
1680 0000D6D8 759B              <1>       jnz    short loc_add_new_cluster_stc_retn
1681                            <1>
1682                            <1> loc_add_new_cluster_update_fat_plc:
1683 0000D6DA A1[448D0100]      <1>       mov    eax, [FAT_anc_LCluster]
1684 0000D6DF 8B0D[488D0100]    <1>       mov    ecx, [FAT_anc_FFCluster]
1685 0000D6E5 E896F9FFFF        <1>       call   update_cluster
1686 0000D6EA 7314              <1>       jnc    short loc_add_new_cluster_save_fat_buffer
1687 0000D6EC 09C0              <1>       or     eax, eax ; EAX = 0 -> cluster value is 0 or eocc
1688 0000D6EE 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 0000D6F0 66BB00FF          <1>       mov    bx, 0FF00h ; recalculate free space (BL = 0)
1695                            <1>                         ; (BH = FFh -> Use ESI as Drv Param. Tbl.)
1696 0000D6F4 50                <1>       push   eax
1697 0000D6F5 E8D8FCFFFF        <1>       call   calculate_fat_freespace
1698 0000D6FA 58                <1>       pop    eax
1699 0000D6FB 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 0000D700 E838FCFFFF          <1>       call   save_fat_buffer
1706 0000D705 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 0000D707 A1[72890100]      <1>       mov    eax, [FAT_ClusterCounter]
1711 0000D70C 66BB01FF          <1>       mov    bx, 0FF01h ; BH = FFh -> ESI -> Dos drv desc. table
1712                            <1>              ; BL = 1 -> add cluster
1713 0000D710 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 0000D712 E8BBFCFFFF        <1>         call    calculate_fat_freespace
1717                            <1>       ;ECX = 0 -> no error, ECX > 0 -> error or invalid return
1718 0000D717 21C9              <1>       and    ecx, ecx ; ECX = 0 -> valid free sector count
1719 0000D719 7409              <1>       jz     short loc_add_new_cluster_return_cluster_number
1720                            <1>
1721                            <1> loc_add_new_cluster_recalc_FAT_freespace:
1722 0000D71B 66BB00FF          <1>       mov    bx, 0FF00h  ; recalculate free space
1723 0000D71F E8AEFCFFFF        <1>         call    calculate_fat_freespace
1724                            <1>       ; cf = 0
1725                            <1> loc_add_new_cluster_return_cluster_number:
1726 0000D724 89C1              <1>       mov    ecx, eax ; Free sector count
1727 0000D726 A1[488D0100]      <1>       mov    eax, [FAT_anc_FFCluster]
1728 0000D72B 0FB65E13          <1>       movzx  ebx, byte [esi+LD_BPB+SecPerClust]
1729                            <1>       ;mov   edi, Cluster_Buffer
1730 0000D72F 31D2              <1>       xor    edx, edx
1731 0000D731 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 0000D732 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 0000D736 807E0300                 <1>       cmp    byte [esi+LD_FATType], 0
1755 0000D73A 761C                     <1>       jna    short write_fs_cluster
1756                                    <1>
1757                                    <1> write_fat_file_sectors:
1758 0000D73C 83E802                   <1>       sub    eax, 2 ; Beginning cluster number is always 2
1759 0000D73F 0FB65613                 <1>       movzx  edx, byte [esi+LD_BPB+BPB_SecPerClust] ; 18/03/2016
1760 0000D743 F7E2                     <1>       mul    edx
1761 0000D745 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 0000D748 E8C14B0000               <1>       call   disk_write
1770 0000D74D 7306                     <1>       jnc    short wclust_retn
1771                                    <1>
1772                                    <1>       ; 15/10/2016 (1Dh -> 18)
1773 0000D74F B812000000               <1>       mov    eax, 18 ; Drive not ready or write error !
1774 0000D754 C3                       <1>       retn
1775                                    <1>
1776                                    <1> wclust_retn:
1777 0000D755 29C0                     <1>       sub    eax, eax ; 0
1778 0000D757 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 0000D758 B980000000               <1>       mov    ecx, 128 ; maximum count of sectors (before eof)
1793 0000D75D E801000000               <1>       call   write_fs_sectors
1794 0000D762 C3                       <1>       retn
1795                                    <1>
1796                                    <1> write_fs_sectors:
1797                                    <1>       ; 21/03/2016 (TRDOS 386 =  TRDOS v2.0)
1798 0000D763 F9                       <1>       stc
1799 0000D764 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 0000D765 807E0301                 <1>       cmp    byte [esi+LD_FATType], 1
1816 0000D769 721E                     <1>         jb     short get_fs_section_by_index
1817                                    <1>
1818 0000D76B 3B4E78                   <1>       cmp    ecx, [esi+LD_Clusters]
1819 0000D76E 7207                     <1>       jb     short gcbi_1
1820                                    <1> gcbi_0:
1821 0000D770 F9                       <1>       stc
1822 0000D771 B823000000               <1>       mov    eax, 23h ; Cluster not available !
1823                                    <1>                       ; MSDOS error code: FCB unavailable
1824 0000D776 C3                       <1>       retn
1825                                    <1> gcbi_1:
1826 0000D777 51                       <1>       push   ecx
1827 0000D778 E8D9F5FFFF               <1>       call   get_next_cluster
1828 0000D77D 59                       <1>       pop    ecx
1829 0000D77E 7203                     <1>       jc     short gcbi_3
1830 0000D780 E2F5                     <1>       loop   gcbi_1
1831                                    <1> gcbi_2:
1832 0000D782 C3                       <1>       retn
1833                                    <1> gcbi_3:
1834 0000D783 09C0                     <1>       or     eax, eax
1835 0000D785 74E9                     <1>       jz     short gcbi_0
1836 0000D787 F5                       <1>       cmc    ; stc
1837 0000D788 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 0000D789 B8FFFFFFFF           <1>       mov    eax, 0FFFFFFFFh
1854 0000D78E 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 0000D78F B800000000           <1>       mov    eax, 0
1872 0000D794 BA00000000           <1>       mov    edx, 0
1873 0000D799 C3                   <1>       retn
3084                                    %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: 24/01/2021
   5                                <1> ; ----------------------------------------------------------------------------
   6                                <1> ; Beginning: 24/01/2016
   7                                <1> ; ----------------------------------------------------------------------------
   8                                <1> ; Assembler: NASM version 2.15 (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>       ; ..........................................................
  45                                <1>       ;
  46                                <1>       ; Retro UNIX 8086 v1 modification:
  47                                <1>       ;      System call number is in EAX register.
  48                                <1>       ;
  49                                <1>       ;      Other parameters are in EDX, EBX, ECX, ESI, EDI, EBP
  50                                <1>       ;     registers depending of function details.
  51                                <1>       ;
  52                                <1>       ; 16/04/2015
  53 0000D79A 368925[5C030300]      <1>         mov     [ss:u.sp], esp ; Kernel stack points to return address
  54                                <1>
  55                                <1>       ; save user registers
  56 0000D7A1 1E                    <1>       push   ds
  57 0000D7A2 06                    <1>       push   es
  58 0000D7A3 0FA0                  <1>       push   fs
  59 0000D7A5 0FA8                  <1>       push   gs
  60 0000D7A7 60                    <1>       pushad  ; eax, ecx, edx, ebx, esp -before pushad-, ebp, esi, edi
  61                                <1>       ;
  62                                <1>       ; ESPACE = [ss:u.sp] - esp ; 4*12 = 48 ; 17/09/2015 ; 06/06/2016
  63                                <1>       ;     (ESPACE is size of space in kernel stack
  64                                <1>       ;     for saving/restoring user registers.)
  65                                <1>       ;
  66 0000D7A8 50                    <1>       push   eax ; 01/07/2015
  67 0000D7A9 66B81000              <1>       mov     ax, KDATA
  68 0000D7AD 8ED8                  <1>         mov     ds, ax
  69 0000D7AF 8EC0                  <1>         mov     es, ax
  70 0000D7B1 8EE0                  <1>         mov     fs, ax
  71 0000D7B3 8EE8                  <1>         mov     gs, ax
  72 0000D7B5 A1[90810100]          <1>       mov    eax, [k_page_dir]
  73 0000D7BA 0F22D8                <1>       mov    cr3, eax
  74 0000D7BD 58                    <1>       pop    eax ; 01/07/2015
```

```
                                <1>       ; 19/10/2015
0000D7BE FC                     <1>       cld
                                <1>       ;
0000D7BF FE05[5B030300]         <1>       inc    byte [sysflg]
                                <1>              ; incb sysflg / indicate a system routine is in progress
0000D7C5 FB                     <1>         sti  ; 18/01/2014
0000D7C6 0F85DF9DFFFF           <1>       jnz     panic ; 24/05/2013
                                <1>              ; beq 1f
                                <1>              ; jmp panic ; / called if trap inside system
                                <1> ;1:
                                <1>       ; 17/03/2017
0000D7CC 80642438FE             <1>       and    byte [esp+ESPACE+8], ~1 ; clear carry flag
                                <1>
                                <1>       ; 16/04/2015
0000D7D1 A3[64030300]           <1>       mov    [u.r0], eax
0000D7D6 8925[60030300]         <1>       mov    [u.usp], esp ; kernel stack points to user's registers
                                <1>
                                <1>       ; 13/01/2017 (TRDOS 386 Feaure only !)
0000D7DC 803D[D4030300]00       <1>       cmp    byte [u.t_lock], 0 ; timer interrupt lock ?
0000D7E3 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
0000D7E9 C1E002                 <1>       shl    eax, 2
                                <1>              ; asl r0 / multiply by 2 to jump indirect in bytes
0000D7EC 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
0000D7F1 F5                     <1>       cmc
0000D7F2 9C                     <1>       pushf
0000D7F3 50                     <1>       push   eax
0000D7F4 8B2D[5C030300]         <1>       mov    ebp, [u.sp] ; Kernel stack at the beginning of sys call
0000D7FA B0FE                   <1>       mov    al, 0FEh ; 11111110b
0000D7FC 1400                   <1>       adc    al, 0 ; al = al + cf
0000D7FE 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
0000D801 5D                     <1>       pop    ebp ; eax
0000D802 9D                     <1>       popf
0000D803 0F8208020000           <1>         jc     badsys
0000D809 A1[64030300]           <1>       mov    eax, [u.r0]
                                <1>       ; system call registers: EAX, EDX, ECX, EBX, ESI, EDI
0000D80E FFA5[14D80000]         <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)
0000D814 [08120100]             <1>       dd sysver    ; 0 ; Get TRDOS 386 version number (v2.0)
0000D818 [73DA0000]             <1>       dd sysexit   ; 1
0000D81C [48DC0000]             <1>       dd sysfork   ; 2
0000D820 [7BE00000]             <1>       dd sysread   ; 3
0000D824 [9AE00000]             <1>       dd syswrite  ; 4
0000D828 [31DE0000]             <1>       dd sysopen   ; 5
0000D82C [52E00000]             <1>       dd sysclose  ; 6
0000D830 [CADB0000]             <1>       dd syswait   ; 7
0000D834 [60DD0000]             <1>       dd syscreat  ; 8
0000D838 [5C200100]             <1>       dd sysrename ; 9  ; TRDOS 386, Rename File (31/12/2017)
0000D83C [D71B0100]             <1>       dd sysdelete ; 10 ; TRDOS 386, Delete File (29/12/2017)
0000D840 [A5050100]             <1>       dd sysexec   ; 11
0000D844 [011D0100]             <1>       dd syschdir  ; 12
0000D848 [C61E0100]             <1>       dd systime   ; 13 ; TRDOS 386, Get Sys Date&Time (30/12/2017)
0000D84C [14E00000]             <1>       dd sysmkdir  ; 14
0000D850 [351D0100]             <1>       dd syschmod  ; 15 ; TRDOS 386, Change Attributes (30/12/2017)
0000D854 [3E1C0100]             <1>       dd sysrmdir  ; 16 ; TRDOS 386, Remove Directory (29/12/2017)
0000D858 [80080100]             <1>       dd sysbreak  ; 17
0000D85C [1B1E0100]             <1>       dd sysdrive  ; 18 ; TRDOS 386, Get/Set Current Drv (30/12/2017)
0000D860 [C1080100]             <1>       dd sysseek   ; 19
0000D864 [D3080100]             <1>       dd systell   ; 20
0000D868 [7D210100]             <1>       dd sysmem    ; 21 ; TRDOS 386, Get Total&Free Mem (31/12/2017)
0000D86C [B3210100]             <1>       dd sysprompt ; 22 ; TRDOS 386, Change Cmd Prompt (31/12/2017)
0000D870 [F5210100]             <1>       dd syspath   ; 23 ; TRDOS 386, Get/Set Run Path (31/12/2017)
0000D874 [62220100]             <1>       dd sysenv    ; 24 ; TRDOS 386, Get/Set Env Vars (31/12/2017)
0000D878 [471F0100]             <1>       dd sysstime  ; 25 ; TRDOS 386, Set Sys Date&Time (30/12/2017)
0000D87C [39090100]             <1>       dd sysquit   ; 26
0000D880 [2D090100]             <1>       dd sysintr   ; 27
0000D884 [6A1E0100]             <1>       dd sysdir    ; 28 ; TRDOS 386, Get Curr Drive&Dir (30/12/2017)
```

```
180 0000D888 [31E10000]          <1>       dd sysemt    ; 29
181 0000D88C [A51E0100]          <1>       dd sysldrvt  ; 30 ; TRDOS 386, Get Logical DOS DDT (30/12/2017)
182 0000D890 [E2E20000]          <1>       dd sysvideo  ; 31 ; TRDOS 386 Video Functions (16/05/2016)
183 0000D894 [2C2C0100]          <1>       dd sysaudio  ; 32 ; TRDOS 386 Audio Functions (16/05/2016)
184 0000D898 [4AE10000]          <1>       dd systimer  ; 33 ; TRDOS 386 Timer Functions (18/05/2016)
185 0000D89C [7A090100]          <1>       dd syssleep  ; 34 ; Retro UNIX 8086 v1 feature only !
186                              <1>                            ; 11/06/2014
187 0000D8A0 [A9090100]          <1>       dd sysmsg    ; 35 ; Retro UNIX 386 v1 feature only !
188                              <1>                            ; 01/07/2015
189 0000D8A4 [C60A0100]          <1>       dd sysgeterr ; 36 ; Retro UNIX 386 v1 feature only !
190                              <1>                            ; 21/09/2015 - get last error number
191 0000D8A8 [AE1B0100]          <1>       dd sysfpstat ; 37 ; TRDOS 386 FPU state option (28/02/2017)
192 0000D8AC [17120100]          <1>       dd syspri    ; 38 ; change priority - TRDOS 386 (20/05/2016)
193 0000D8B0 [86D90000]          <1>       dd sysrele   ; 39 ; TRDOS 386 (19/05/2016) (0 -> 39)
194 0000D8B4 [4A130100]          <1>       dd sysfff    ; 40 ; Find First File - TRDOS 386 (15/10/2016)
195 0000D8B8 [29140100]          <1>       dd sysfnf    ; 41 ; Find Next File - TRDOS 386 (15/10/2016)
196 0000D8BC [991A0100]          <1>       dd sysalloc  ; 42 ; Allocate contiguous memory block/pages
197                              <1>                            ; TRDOS 386 (19/02/2017) DMA buff fuctions
198 0000D8C0 [571B0100]          <1>       dd sysdalloc ; 43 ; Deallocate contiguous memory block/pages
199                              <1>                            ; TRDOS 386 (19/02/2017) DMA buff fuctions
200 0000D8C4 [921B0100]          <1>       dd syscalbac ; 44 ; IRQ Callback and Signal Response Byte
201                              <1>                            ; service setup - TRDOS 386 (20/02/2017)
202                              <1>                            ; 28/08/2017 (20/08/2017)
203 0000D8C8 [BD340100]          <1>       dd sysdma    ; 45 ; TRDOS 386 - (ISA) DMA service
204                              <1>
205                              <1> end_of_syscalls:
206                              <1>
207                              <1> error:
208                              <1>       ; 18/05/2016
209                              <1>       ; 13/05/2016
210                              <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
211                              <1>       ; 16/04/2015 - 17/09/2015 (Retro UNIX 386 v1)
212                              <1>       ; 10/04/2013 - 07/08/2013 (Retro UNIX 8086 v1)
213                              <1>       ;
214                              <1>       ; 'error' merely sets the error bit off the processor status (c-bit)
215                              <1>       ; then falls right into the 'sysret', 'sysrele' return sequence.
216                              <1>       ;
217                              <1>       ; INPUTS -> none
218                              <1>       ; OUTPUTS ->
219                              <1>       ;     processor status - carry (c) bit is set (means error)
220                              <1>       ;
221                              <1>       ; 26/05/2013 (Stack pointer must be reset here!
222                              <1>       ;           Because, jumps to error procedure
223                              <1>       ;           disrupts push-pop nesting balance)
224                              <1>       ;
225 0000D8CC 8B2D[5C030300]      <1>       mov    ebp, [u.sp] ; interrupt (system call) return (iretd) address
226 0000D8D2 804D0801            <1>       or     byte [ebp+8], 1  ; set carry bit of flags register
227                              <1>                               ; (system call will return with cf = 1)
228                              <1>             ; bis $1,20.(r1) / set c bit in processor status word below
229                              <1>                               ; / users stack
230                              <1>       ; 17/09/2015
231 0000D8D6 83ED30              <1>       sub    ebp, ESPACE ; 48 ; total size of stack frame ('sysdefs.inc')
232                              <1>                               ; for saving/restoring user registers
233                              <1>       ;cmp   ebp, [u.usp]
234                              <1>       ;je    short err0
235 0000D8D9 892D[60030300]      <1>       mov    [u.usp], ebp
236                              <1> ;err0:
237                              <1>       ; 01/09/2015
238 0000D8DF 8B25[60030300]      <1>       mov    esp, [u.usp]     ; Retro Unix 8086 v1 modification!
239                              <1>                               ; 10/04/2013
240                              <1>                               ; (If an I/O error occurs during disk I/O,
241                              <1>                               ; related procedures will jump to 'error'
242                              <1>                               ; procedure directly without returning to
243                              <1>                               ; the caller procedure. So, stack pointer
244                              <1>                                     ; must be restored here.)
245                              <1>       ; 13/05/2016
246                              <1>       ; NOTE: (The last) error code is in 'u.error', it can be retrieved by
247                              <1>       ;     'get last error' system call later.
248                              <1>
249                              <1>       ; 03/09/2015 - 09/06/2015 - 07/08/2013
250 0000D8E5 C605[C6030300]00    <1>       mov    byte [u.kcall], 0 ; namei_r, mkdir_w reset
251                              <1>
252                              <1> sysret: ; < return from system call>
253                              <1>       ; 01/03/2017
254                              <1>       ; 28/02/2017
255                              <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
256                              <1>       ; 16/04/2015 - 10/09/2015 (Retro UNIX 386 v1)
257                              <1>       ; 10/04/2013 - 23/02/2014 (Retro UNIX 8086 v1)
258                              <1>       ;
259                              <1>       ; 'sysret' first checks to see if process is about to be
260                              <1>       ; terminated (u.bsys). If it is, 'sysexit' is called.
261                              <1>       ; If not, following happens:
262                              <1>       ;     1) The user's stack pointer is restored.
263                              <1>       ;     2) r1=0 and 'iget' is called to see if last mentioned
264                              <1>       ;        i-node has been modified. If it has, it is written out
265                              <1>       ;        via 'ppoke'.
266                              <1>       ;     3) If the super block has been modified, it is written out
267                              <1>       ;        via 'ppoke'.
268                              <1>       ;     4) If the dismountable file system's super block has been
269                              <1>       ;        modified, it is written out to the specified device
270                              <1>       ;        via 'ppoke'.
271                              <1>       ;     5) A check is made if user's time quantum (uquant) ran out
272                              <1>       ;        during his execution. If so, 'tswap' is called to give
273                              <1>       ;        another user a chance to run.
274                              <1>       ;     6) 'sysret' now goes into 'sysrele'.
275                              <1>       ;         (See 'sysrele' for conclusion.)
276                              <1>       ;
277                              <1>       ; Calling sequence:
278                              <1>       ;     jump table or 'br sysret'
279                              <1>       ; Arguments:
280                              <1>       ;     -
281                              <1>       ; ..........................................................
282                              <1>       ;
283                              <1>       ; ((AX=r1 for 'iget' input))
284                              <1>       ;
```

```
0000D8EC 31C0                <1>       xor    eax, eax ; 28/02/2017
                             <1> sysret0: ; 29/07/2015 (eax = 0, jump from sysexec)
0000D8EE FEC0                <1>       inc    al ; 04/05/2013
0000D8F0 3805[B2030300]      <1>       cmp    [u.bsys], al ; 1
                             <1>              ; tstb u.bsys / is a process about to be terminated because
0000D8F6 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
0000D8FC FEC8                <1>       dec    al ; mov ax, 0
                             <1>              ; clr r1 / zero r1 to check last mentioned i-node
0000D8FE E80A4A0000          <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:
0000D903 803D[A8030300]00    <1>       cmp    byte [u.quant], 0 ; 16/05/2013
                             <1>              ; tstb uquant / is the time quantum 0?
0000D90A 7705                <1>         ja      short swapret
                             <1>              ; bne 1f / no, don't swap it out
                             <1> sysrelease: ; 07/12/2013 (jump from 'clock')
0000D90C E8BF370000          <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
0000D911 E8F7490000          <1>       call   isintr
                             <1>       ; 20/10/2013
0000D916 7405                <1>       jz     short sysrel1
0000D918 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:
0000D91D FA                  <1>       cli    ; 14/10/2015
                             <1> sysrel2:
                             <1>       ; 28/02/2017
                             <1>       ; Check if there is a (delayed) callback for current user/process
0000D91E A0[D7030300]        <1>       mov    al, [u.irqwait]
0000D923 240F                <1>       and    al, 0Fh ; is there a waiting IRQ callback service ?
0000D925 7444                <1>       jz     short sysrel8 ; no
                             <1>
                             <1>       ; Set return to IRQ callback service and return from the service
0000D927 0FB6D8              <1>       movzx  ebx, al
0000D92A 883D[D7030300]      <1>       mov    [u.irqwait], bh ; 0 ; reset
0000D930 8A9B[58410100]      <1>       mov    bl, [ebx+IRQenum] ; (available) IRQ index +1 (1 to 9)
                             <1>       ; 01/03/2017
0000D936 FECB                <1>       dec    bl ; IRQ index number, 0 to 8
0000D938 7831                <1>       js     short sysrel8 ; 0 -> FFh (not in use!?)
                             <1>       ;
0000D93A A0[B3030300]        <1>       mov    al, [u.uno] ; current process (user) number
0000D93F 3883[C2930100]      <1>       cmp    [ebx+IRQ.owner], al
0000D945 7524                <1>       jne    short sysrel8 ; it is not the current user/process !?
0000D947 F683[D4930100]01    <1>       test   byte [ebx+IRQ.method], 1 ; callback ?
0000D94E 741B                <1>       jz     short sysrel8 ; not a callback method !?
                             <1>
0000D950 8B93[E6930100]      <1>       mov    edx, [ebx+IRQ.addr] ; IRQ callback service address (virtual)
0000D956 C605[D8030300]01    <1>       mov    byte [u.r_lock], 1 ; IRQ callback service in progress flag
                             <1>
0000D95D E816380000          <1>       call   wswap ; save user's registers & status
                             <1>                    ;      (for return from IRQ callback service)
                             <1>
0000D962 8B2D[5C030300]      <1>       mov    ebp, [u.sp]; kernel's stack, points to EIP (user)
0000D968 895500              <1>       mov    [ebp], edx ; IRQ call back service address
                             <1> sysrel8:
0000D96B FE0D[5B030300]      <1>       dec    byte [sysflg]
                             <1>              ; decb sysflg / turn system flag off
                             <1>
0000D971 A1[B8030300]        <1>       mov    eax, [u.pgdir]
0000D976 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
0000D979 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>       ;
0000D97A A1[64030300]           <1>       mov    eax, [u.r0]  ; ((return value in EAX))
0000D97F 0FA9                   <1>       pop    gs
0000D981 0FA1                   <1>       pop    fs
0000D983 07                     <1>       pop    es
0000D984 1F                     <1>       pop    ds
                                <1>       ;or    word [esp+8], 200h ; 22/01/2017 ; force enabling interrupts
0000D985 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
0000D986 803D[D4030300]00       <1>       cmp    byte [u.t_lock], 0 ; TIMER INT LOCK
                                <1>       ;je    short sysrel0 ; classic (Retro UNIX 386 type) sysrele
0000D98D 7734                   <1>       ja     short sysrel3
                                <1>       ; 27/02/2017
0000D98F 803D[D8030300]00       <1>       cmp    byte [u.r_lock], 0 ; IRQ callback lock
0000D996 0F8667FFFFFF           <1>       jna    sysrel0 ; classic sysrele ; 24/03/2017
0000D99C E859000000             <1>       call   sysrel7
0000D9A1 803D[D8030300]00       <1>       cmp    byte [u.r_lock], 0 ; IRQ callback service lock
0000D9A8 7628                   <1>       jna    short sysrel4
0000D9AA C605[D8030300]00       <1>       mov    byte [u.r_lock], 0 ; reset
                                <1>       ;mov   byte [u.irqwait], 0 ; reset ; 28/02/2017
0000D9B1 A0[D9030300]           <1>       mov    al, [u.r_mode]
0000D9B6 08C0                   <1>       or     al, al
0000D9B8 7518                   <1>       jnz    short sysrel4
0000D9BA FEC8                   <1>       dec    al
0000D9BC A2[D9030300]           <1>       mov    [u.r_mode], al ; 0FFh ; not necessary !?
0000D9C1 EB32                   <1>       jmp    short sysrel6
                                <1> sysrel3:
                                <1>       ; 27/02/2017
0000D9C3 E832000000             <1>       call   sysrel7
                                <1>       ; 14/01/2017
0000D9C8 28C0                   <1>       sub    al, al
0000D9CA 3805[D4030300]         <1>       cmp    [u.t_lock], al ; 0 ; TIMER INT LOCK
0000D9D0 770E                   <1>       ja     short sysrel5 ; yes
                                <1> sysrel4:
                                <1>       ; 29/01/2017
0000D9D2 8B44241C               <1>       mov    eax, [esp+28] ; eax
0000D9D6 A3[64030300]           <1>       mov    [u.r0], eax
0000D9DB E93EFFFFFF             <1>       jmp    sysrel2
                                <1> sysrel5:
0000D9E0 A2[D4030300]           <1>       mov    [u.t_lock], al ; 0 ; reset
0000D9E5 A0[D5030300]           <1>       mov    al, [u.t_mode]
0000D9EA 20C0                   <1>       and    al, al
                                <1>       ;jnz   short sysrel2 ; 0FFh ; user mode
0000D9EC 75E4                   <1>       jnz    short sysrel4 ; 29/01/2017
0000D9EE FEC8                   <1>       dec    al
0000D9F0 A2[D5030300]           <1>       mov    [u.t_mode], al ; 0FFh ; not necessary !?
                                <1> sysrel6:
                                <1>       ; cpu will continue from the interrupted sytem call addr
0000D9F5 61                     <1>       popad        ; edi, esi, ebp, esp, ebx, edx, ecx, eax
0000D9F6 83C410                 <1>       add    esp, 16      ; pass segment segisters: ds, es, fs, gs
0000D9F9 CF                     <1>       iretd        ; eip, cs, eflags
                                <1>
                                <1> sysrel7:
0000D9FA 0FB61D[B3030300]       <1>       movzx  ebx, byte [u.uno] ; current process number
0000DA01 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
0000DA05 8B83[BC000300]         <1>       mov     eax, [ebx+p.upage-4] ; UPAGE address
0000DA0B FA                     <1>       cli    ; disable interrupts till 'iretd'
0000DA0C E99F370000             <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>       ;
0000DA11 FE05[B2030300]         <1>       inc    byte [u.bsys]
                                <1>       ;
0000DA17 8B1D[5C030300]         <1>       mov    ebx, [u.sp] ; esp at the beginning of 'sysent'
0000DA1D 8B03                   <1>       mov    eax, [ebx] ; EIP (return address, not 'INT 30h' address)
0000DA1F 83E802                 <1>       sub    eax, 2 ; CDh, ##h
```

```
0000DA22 E8DC68FFFF          <1>       call   dwordtohex
0000DA27 8915[313F0100]      <1>       mov    [eip_str], edx
0000DA2D A3[353F0100]        <1>       mov    [eip_str+4], eax
0000DA32 A1[64030300]        <1>       mov    eax, [u.r0]
0000DA37 E8C768FFFF          <1>       call   dwordtohex
0000DA3C 8915[203F0100]      <1>       mov    [eax_str], edx
0000DA42 A3[243F0100]        <1>       mov    [eax_str+4], eax
                             <1>
0000DA47 66C705[153F0100]34- <1>       mov    word [int_num_str], SYSCALL_INT_NUM ; 25/12/2016
0000DA4F 30                  <1>
                             <1>
0000DA50 BE[E73E0100]        <1>       mov    esi, ifc_msg ; "invalid funtion call !" msg (trdosk9.s)
0000DA55 E8EB9AFFFF          <1>       call   print_msg
                             <1>
0000DA5A 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
0000DA5C FB                  <1>       sti
                             <1>       ; 07/12/2013
0000DA5D 66FF05[AC030300]    <1>       inc    word [u.quit]
0000DA64 7408                <1>       jz     short intrct0 ; FFFFh -> 0
0000DA66 66FF0D[AC030300]    <1>       dec    word [u.quit]
                             <1>       ; 16/04/2015
0000DA6D C3                  <1>       retn
                             <1> intrct0:
0000DA6E 58                  <1>       pop    eax ; call intract -> retn
                             <1>       ;
0000DA6F 31C0                <1>       xor    eax, eax
0000DA71 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
0000DA73 6648                   <1>       dec    ax ; 0
0000DA75 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
0000DA7B FA                     <1>       cli    ; disable interrupts
                                <1>       ; 23/01/2017 - reset timer frequency (to 18.2Hz)
0000DA7C B036                   <1>       mov    al, 00110110b ; 36h
0000DA7E E643                   <1>       out    43h, al
0000DA80 28C0                   <1>       sub    al, al ; 0
0000DA82 E640                   <1>       out    40h, al ; LB
0000DA84 E640                   <1>       out    40h, al ; HB
                                <1>       ;
0000DA86 0FB61D[B3030300]       <1>       movzx  ebx, byte [u.uno]
                                <1>       ;mov   bl, [u.uno] ; process number of dying process
0000DA8D 3883[FF000300]         <1>       cmp    byte [ebx+p.timer-1], al ; 0
0000DA93 763A                   <1>       jna    short sysexit_12 ; no timer events for this process
0000DA95 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
0000DA9B 8A0D[238E0100]         <1>       mov    cl, [timer_events] ; 14/11/2017
                                <1>       ;cli   ; disable interrupts
0000DAA1 B410                   <1>       mov    ah, 16 ; number of available timer events
0000DAA3 BE[60040300]           <1>       mov    esi, timer_set ; beginning address of timer events
                                <1> sysexit_7:
0000DAA8 8A06                   <1>       mov    al, [esi] ; process number (of timer event)
0000DAAA 38D8                   <1>       cmp    al, bl ; process number comparison
0000DAAC 7411                   <1>       je     short sysexit_10
0000DAAE 20C0                   <1>       and    al, al
0000DAB0 7404                   <1>       jz     short sysexit_9
                                <1> sysexit_8:
0000DAB2 FEC9                   <1>       dec    cl
0000DAB4 7416                   <1>       jz     short sysexit_11
                                <1> sysexit_9:
0000DAB6 FECC                   <1>       dec    ah
0000DAB8 7415                   <1>       jz     short sysexit_12
0000DABA 83C610                 <1>       add    esi, 16
0000DABD EBE9                   <1>       jmp    short sysexit_7
                                <1>
                                <1> sysexit_10:
                                <1>       ;mov   byte [esi], 0
0000DABF 66C7060000             <1>       mov    word [esi], 0
                                <1>       ;mov   dword [esi+12], 0
                                <1>       ;
0000DAC4 FE0D[238E0100]         <1>       dec    byte [timer_events] ; 02/01/2017
                                <1>       ;
0000DACA EBE6                   <1>       jmp    short sysexit_8
                                <1>
                                <1> sysexit_11:
0000DACC 6629C0                 <1>       sub    ax, ax ; 0 ; 26/02/2017
                                <1> sysexit_12:
                                <1>       ; 26/02/2017 (Unlink IRQ callbacks belong to the user)
0000DACF 803D[D6030300]00       <1>       cmp    byte [u.irqc], 0 ; Count of IRQ callbacks
0000DAD6 7E2E                   <1>       jng    short sysexit_16 ; zero or invalid
                                <1>       ; 28/02/2017
                                <1>       ; clear IRQ callback flags (for 'sysrele' and 'sysret')
0000DAD8 A2[D7030300]           <1>       mov    [u.irqwait], al ; 0 ; force to clear waiting flag
0000DADD A2[D8030300]           <1>       mov    [u.r_lock], al ; 0 ; force to clear busy flag
0000DAE2 BE[C2930100]           <1>       mov    esi, IRQ.owner
                                <1> sysexit_13:
0000DAE7 AC                     <1>       lodsb
0000DAE8 3A05[B3030300]         <1>       cmp    al, [u.uno] ; owner = current user ?
0000DAEE 750C                   <1>       jne    short sysexit_14
0000DAF0 C646FF00               <1>       mov    byte [esi-1], 0 ; owner = 0 : Free
0000DAF4 FE0D[D6030300]         <1>       dec    byte [u.irqc]
0000DAFA 7408                   <1>       jz     short sysexit_15
                                <1> sysexit_14:
0000DAFC 81FE[CA930100]         <1>       cmp    esi, IRQ.owner + 8 ; the last IRQ index number ?
0000DB02 76E3                   <1>       jna    short sysexit_13 ; no
                                <1> sysexit_15:
0000DB04 30C0                   <1>       xor    al, al ; 0
                                <1> sysexit_16: ; 2:
```

```
0000DB06 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
0000DB07 E8912C0000         <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
0000DB0C FEC0               <1>       inc    al
                            <1>             ; inc r1 / increment file descriptor
                            <1>       ;cmp   ax, 10
0000DB0E 3C0A               <1>       cmp    al, 10
                            <1>             ; cmp r1,$10. / end of u.fp list?
0000DB10 72F5               <1>       jb     short sysexit_1
                            <1>             ; blt 1b / no, go back
                            <1>       ;movzx ebx, byte [u.uno]
0000DB12 8A1D[B3030300]     <1>       mov    bl, [u.uno] ; 02/01/2017
                            <1>             ; movb u.uno,r1 / yes, move dying process's number to r1
0000DB18 88A3[AF000300]     <1>       mov    [ebx+p.stat-1], ah ; 0, SFREE
                            <1>             ; clrb p.stat-1(r1) / free the process
                            <1>       ; 10/04/2017
0000DB1E 381D[39940100]     <1>       cmp    [audio_user], bl
0000DB24 7518               <1>       jne    short sysexit_17
                            <1>       ; reset audio device (current) owner and 'initializated' flag
0000DB26 883D[39940100]     <1>       mov    [audio_user], bh ; 0
                            <1>       ; 27/05/2017
0000DB2C 8B0D[24940100]     <1>       mov    ecx,  [audio_buffer]
0000DB32 09C9               <1>       or     ecx, ecx
0000DB34 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)
0000DB36 29C9               <1>       sub    ecx, ecx
0000DB38 890D[24940100]     <1>       mov    [audio_buffer], ecx ; 0
                            <1>       ;pop   ebx
                            <1> sysexit_17:
                            <1>       ;shl   bx, 1
0000DB3E D0E3               <1>       shl    bl, 1
                            <1>             ; asl r1 / use r1 for index into the below tables
0000DB40 668B8B[1E000300]   <1>       mov    cx, [ebx+p.pid-2]
                            <1>             ; mov p.pid-2(r1),r3 / move dying process's name to r3
0000DB47 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
0000DB4E 30DB               <1>       xor    bl, bl ; 0
                            <1>             ; clr r2
0000DB50 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
0000DB52 80C302             <1>       add    bl, 2
                            <1>             ; add $2,r2 / search parent process table
                            <1>                       ; / for dying process's name
0000DB55 66398B[3E000300]   <1>       cmp    [ebx+p.ppid-2], cx
                            <1>             ; cmp p.ppid-2(r2),r3 / found it?
0000DB5C 7513               <1>       jne    short sysexit_4
                            <1>             ; bne 3f / no
                            <1>       ;shr   bx, 1
0000DB5E D0EB               <1>       shr    bl, 1
                            <1>             ; asr r2 / yes, it is a parent
0000DB60 80BB[AF000300]03   <1>       cmp    byte [ebx+p.stat-1], 3 ; SZOMB
                            <1>             ; cmpb p.stat-1(r2),$3 / is the child of this
                            <1>                               ; / dying process a zombie
0000DB67 7506               <1>       jne    short sysexit_3
                            <1>             ; bne 2f / no
0000DB69 88A3[AF000300]     <1>       mov    [ebx+p.stat-1], ah ; 0, SFREE
                            <1>             ; clrb p.stat-1(r2) / yes, free the child process
                            <1> sysexit_3: ; 2:
                            <1>       ;shr   bx, 1
0000DB6F D0E3               <1>       shl    bl, 1
                            <1>             ; asl r2
                            <1> sysexit_4: ; 3:
                            <1>             ; / search the process name table
                            <1>             ; / for the dying process's parent
0000DB71 663993[1E000300]   <1>       cmp    [ebx+p.pid-2], dx
                            <1>             ; cmp p.pid-2(r2),r4 / found it?
0000DB78 7502               <1>       jne    short sysexit_5
                            <1>             ; bne 3f / no
0000DB7A 89DE               <1>       mov    esi, ebx
                            <1>             ; mov r2,r5 / yes, put index to p.pid table (parents
                            <1>                       ; / process # x2) in r5
                            <1> sysexit_5: ; 3:
                            <1>       ;cmp   bx, nproc + nproc
0000DB7C 80FB20             <1>       cmp    bl, nproc + nproc
                            <1>             ; cmp r2,$nproc+nproc / has whole table been searched?
0000DB7F 72D1               <1>       jb     short sysexit_2
                            <1>             ; blt 1b / no, go back
                            <1>             ; mov r5,r1 / yes, r1 now has parents process # x2
0000DB81 21F6               <1>       and    esi, esi ; r5=r1
0000DB83 7436               <1>       jz     short sysexit_6
                            <1>             ; beq 2f / no parent has been found.
                            <1>                    ; / The process just dies
0000DB85 66D1EE             <1>       shr    si, 1
                            <1>             ; asr r1 / set up index to p.stat
0000DB88 8A86[AF000300]     <1>       mov    al, [esi+p.stat-1]
                            <1>             ; movb p.stat-1(r1),r2 / move status of parent to r2
```

```
0000DB8E 20C0               <1>       and   al, al
0000DB90 7429               <1>       jz    short sysexit_6
                            <1>             ; beq 2f / if its been freed, 2f
0000DB92 3C03               <1>       cmp   al, 3
                            <1>             ; cmp r2,$3 / is parent a zombie?
0000DB94 7425               <1>       je    short sysexit_6
                            <1>             ; beq 2f / yes, 2f
                            <1>       ; BH = 0
0000DB96 8A1D[B3030300]     <1>       mov   bl, [u.uno]
                            <1>             ; movb u.uno,r3 / move dying process's number to r3
0000DB9C C683[AF000300]03   <1>       mov   byte [ebx+p.stat-1], 3  ; SZOMB
                            <1>             ; movb $3,p.stat-1(r3) / make the process a zombie
0000DBA3 3C01               <1>       cmp   al, 1 ; SRUN
0000DBA5 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
0000DBA7 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)
0000DBAE 6689F0             <1>       mov   ax, si ; r1  (process number in AL)
                            <1>       ;
                            <1>       ;mov  ebx, runq + 4
                            <1>             ; mov $runq+4,r2 / on the runq
0000DBB1 BB[54030300]       <1>       mov   ebx, runq+2 ; normal run queue ; 02/01/2017
0000DBB6 E82D360000         <1>       call  putlu
                            <1>             ; jsr r0, putlu
                            <1> sysexit_6:
                            <1>             ; / the process dies
0000DBBB C605[B3030300]00   <1>       mov   byte [u.uno], 0
                            <1>             ; clrb u.uno / put zero as the process number,
                            <1>                 ; / so "swap" will
0000DBC2 E823350000         <1>       call  swap
                            <1>             ; jsr r0,swap / overwrite process with another process
                            <1> hlt_sys:
                            <1>       ;sti
                            <1> hlts0:
0000DBC7 F4                 <1>       hlt
0000DBC8 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:
0000DBCA 0FB61D[B3030300]   <1>       movzx  ebx, byte [u.uno] ; 01/09/2015
                            <1>              ; movb u.uno,r1 / put parents process number in r1
0000DBD1 D0E3               <1>       shl    bl, 1
                            <1>       ;shl   bx, 1
                            <1>              ; asl r1 / x2 to get index into p.pid table
0000DBD3 668B83[1E000300]   <1>       mov    ax, [ebx+p.pid-2]
                            <1>              ; mov p.pid-2(r1),r1 / get the name of this process
0000DBDA 31F6               <1>       xor    esi, esi
                            <1>              ; clr r2
0000DBDC 31C9               <1>       xor    ecx, ecx ; 30/10/2013
                            <1>       ;xor   cl, cl
                            <1>              ; clr r3 / initialize reg 3
                            <1> syswait_1: ; 1:
0000DBDE 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
0000DBE2 663B86[3E000300]   <1>       cmp    ax, [esi+p.ppid-2]
                            <1>              ; cmp p.ppid-2(r2),r1 / r2 will contain the childs
                            <1>                                   ; / process number
0000DBE9 7535               <1>       jne    short syswait_3
```

```
                                <1>             ;bne 3f / branch if no match of parent process name
                                <1>         ;inc   cx
0000DBEB FEC1                   <1>         inc    cl
                                <1>             ;inc r3 / yes, a match, r3 indicates number of children
0000DBED 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).
0000DBF0 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?
0000DBF7 7524                   <1>         jne    short syswait_2
                                <1>             ; bne 2f / no, skip it
0000DBF9 88BE[AF000300]         <1>         mov    [esi+p.stat-1], bh ; 0
                                <1>             ; clrb p.stat-1(r2) / yes, free it
0000DBFF 66D1E6                 <1>         shl    si, 1
                                <1>             ; asl r2 / r2x2 to get index into p.pid table
0000DC02 0FB786[1E000300]       <1>         movzx  eax, word [esi+p.pid-2]
0000DC09 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>         ;
0000DC0E 6629C0                 <1>         sub    ax, ax
0000DC11 668986[3E000300]       <1>         mov    [esi+p.ppid-2], ax ; 0 ; 17/09/2015
0000DC18 E9D1FCFFFF             <1>         jmp    sysret0 ; ax = 0
                                <1>         ;
                                <1>         ;jmp   sysret
                                <1>                ; br sysret1 / return cause child is dead
                                <1> syswait_2: ; 2:
0000DC1D 66D1E6                 <1>         shl    si, 1
                                <1>                ; asl r2 / r2x2 to get index into p.ppid table
                                <1> syswait_3: ; 3:
0000DC20 6683FE20               <1>         cmp    si, nproc+nproc
                                <1>                ; cmp r2,$nproc+nproc / have all processes been checked?
0000DC24 72B8                   <1>         jb     short syswait_1
                                <1>                ; blt 1b / no, continue search
                                <1>         ;and   cx, cx
0000DC26 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
0000DC28 750B                   <1>         jnz    short syswait_4
                                <1>         ;jz    error
                                <1>                ; beq error1 / there are no children, error
0000DC2A 890D[64030300]         <1>         mov    [u.r0], ecx ; 0
0000DC30 E997FCFFFF             <1>         jmp    error
                                <1> syswait_4:
0000DC35 8A1D[B3030300]         <1>         mov    bl, [u.uno]
                                <1>                ; movb u.uno,r1 / there are children so put
                                <1>                             ; / parent process number in r1
0000DC3B 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
0000DC41 E8A4340000             <1>         call   swap
                                <1>                ; jsr r0,swap / swap it out, because it's waiting
0000DC46 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
                                <1>         ; 28/08/2015, 01/09/2015, 02/09/2015
                                <1>         ; 09/05/2015, 10/05/2015, 14/05/2015
                                <1>         ; 06/05/2015 (Retro UNIX 386 v1 - Beginning)
                                <1>         ; 24/05/2013 - 14/02/2014 (Retro UNIX 8086 v1)
                                <1>         ;
                                <1>         ; 'sysfork' creates a new process. This process is referred
                                <1>         ; to as the child process. This new process core image is
                                <1>         ; a copy of that of the caller of 'sysfork'. The only
                                <1>         ; distinction is the return location and the fact that (u.r0)
                                <1>         ; in the old process (parent) contains the process id (p.pid)
                                <1>         ; of the new process (child). This id is used by 'syswait'.
                                <1>         ; 'sysfork' works in the following manner:
                                <1>         ;    1) The process status table (p.stat) is searched to find
                                <1>         ;      a process number that is unused. If none are found
                                <1>         ;      an error occurs.
                                <1>         ;    2) when one is found, it becomes the child process number
                                <1>         ;      and it's status (p.stat) is set to active.
                                <1>         ;    3) If the parent had a control tty, the interrupt
                                <1>         ;      character in that tty buffer is cleared.
                                <1>         ;    4) The child process is put on the lowest priority run
                                <1>         ;      queue via 'putlu'.
                                <1>         ;    5) A new process name is gotten from 'mpid' (actually
                                <1>         ;      it is a unique number) and is put in the child's unique
                                <1>         ;      identifier; process id (p.pid).
                                <1>         ;    6) The process name of the parent is then obtained and
                                <1>         ;      placed in the unique identifier of the parent process
                                <1>         ;      name is then put in 'u.r0'.
                                <1>         ;    7) The child process is then written out on disk by
                                <1>         ;      'wswap',i.e., the parent process is copied onto disk
                                <1>         ;      and the child is born. (The child process is written
```

```
1019                                  <1>       ;      out on disk/drum with 'u.uno' being the child process
1020                                  <1>       ;      number.)
1021                                  <1>       ;    8) The parent process number is then restored to 'u.uno'.
1022                                  <1>       ;    9) The child process name is put in 'u.r0'.
1023                                  <1>       ;   10) The pc on the stack sp + 18 is incremented by 2 to
1024                                  <1>       ;      create the return address for the parent process.
1025                                  <1>       ;   11) The 'u.fp' list as then searched to see what files
1026                                  <1>       ;      the parent has opened. For each file the parent has
1027                                  <1>       ;      opened, the corresponding 'fsp' entry must be updated
1028                                  <1>       ;      to indicate that the child process also has opened
1029                                  <1>       ;      the file. A branch to 'sysret' is then made.
1030                                  <1>       ;
1031                                  <1>       ; Calling sequence:
1032                                  <1>       ;      from shell ?
1033                                  <1>       ; Arguments:
1034                                  <1>       ;      -
1035                                  <1>       ; Inputs: -
1036                                  <1>       ; Outputs: *u.r0 - child process name
1037                                  <1>       ; ............................................................
1038                                  <1>       ;
1039                                  <1>       ; Retro UNIX 8086 v1 modification:
1040                                  <1>       ;      AX = r0 = PID (>0) (at the return of 'sysfork')
1041                                  <1>       ;      = process id of child a parent process returns
1042                                  <1>       ;      = process id of parent when a child process returns
1043                                  <1>       ;
1044                                  <1>       ;        In original UNIX v1, sysfork is called and returns as
1045                                  <1>       ;      in following manner: (with an example: c library, fork)
1046                                  <1>       ;
1047                                  <1>       ;      1:
1048                                  <1>       ;            sys    fork
1049                                  <1>       ;                   br 1f  / child process returns here
1050                                  <1>       ;            bes    2f     / parent process returns here
1051                                  <1>       ;            / pid of new process in r0
1052                                  <1>       ;            rts    pc
1053                                  <1>       ;      2: / parent process condionally branches here
1054                                  <1>       ;            mov    $-1,r0 / pid = -1 means error return
1055                                  <1>       ;            rts    pc
1056                                  <1>       ;
1057                                  <1>       ;      1: / child process brances here
1058                                  <1>       ;            clr    r0   / pid = 0 in child process
1059                                  <1>       ;            rts    pc
1060                                  <1>       ;
1061                                  <1>       ;      In UNIX v7x86 (386) by Robert Nordier (1999)
1062                                  <1>       ;            // pid = fork();
1063                                  <1>       ;            //
1064                                  <1>       ;            // pid == 0 in child process;
1065                                  <1>       ;            // pid == -1 means error return
1066                                  <1>       ;            // in child,
1067                                  <1>       ;            //     parents id is in par_uid if needed
1068                                  <1>       ;
1069                                  <1>       ;            _fork:
1070                                  <1>       ;                  mov    $.fork,eax
1071                                  <1>       ;                  int    $0x30
1072                                  <1>       ;                  jmp    1f
1073                                  <1>       ;                  jnc    2f
1074                                  <1>       ;                  jmp    cerror
1075                                  <1>       ;            1:
1076                                  <1>       ;                  mov    eax,_par_uid
1077                                  <1>       ;                  xor    eax,eax
1078                                  <1>       ;            2:
1079                                  <1>       ;                  ret
1080                                  <1>       ;
1081                                  <1>       ;      In Retro UNIX 8086 v1,
1082                                  <1>       ;      'sysfork' returns in following manner:
1083                                  <1>       ;
1084                                  <1>       ;            mov    ax, sys_fork
1085                                  <1>       ;            mov    bx, offset @f ; routine for child
1086                                  <1>       ;            int    20h
1087                                  <1>       ;            jc     error
1088                                  <1>       ;
1089                                  <1>       ;      ; Routine for parent process here (just after 'jc')
1090                                  <1>       ;            mov    word ptr [pid_of_child], ax
1091                                  <1>       ;            jmp    next_routine_for_parent
1092                                  <1>       ;
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 0000DC48 31F6                   <1>       xor    esi, esi
1107                                  <1>              ; clr r1
1108                                  <1> sysfork_1: ; 1: / search p.stat table for unused process number
1109 0000DC4A 46                     <1>       inc    esi
1110                                  <1>              ; inc r1
1111 0000DC4B 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 0000DC52 760B                   <1>       jna    short sysfork_2
1114                                  <1>              ; beq 1f / it's unused so branch
1115 0000DC54 6683FE10               <1>       cmp    si, nproc
1116                                  <1>              ; cmp r1,$nproc / all processes checked
1117 0000DC58 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 0000DC5A E96DFCFFFF       <1>       jmp   error
1130                            <1> sysfork_2: ; 1:
1131 0000DC5F E8657FFFFF       <1>       call  allocate_page
1132 0000DC64 0F8262FCFFFF     <1>       jc    error
1133 0000DC6A 50               <1>       push  eax   ; UPAGE (user structure page) address
1134                            <1>       ; Retro UNIX 386 v1 modification!
1135 0000DC6B E86881FFFF       <1>       call  duplicate_page_dir
1136                            <1>             ; EAX = New page directory
1137 0000DC70 730B             <1>       jnc   short sysfork_3
1138 0000DC72 58               <1>       pop   eax   ; UPAGE (user structure page) address
1139 0000DC73 E82F81FFFF       <1>       call  deallocate_page
1140 0000DC78 E94FFCFFFF       <1>       jmp   error
1141                            <1> sysfork_3:
1142                            <1>       ; Retro UNIX 386 v1 modification !
1143 0000DC7D 56               <1>       push  esi
1144 0000DC7E E8F5340000       <1>       call  wswap ; save current user (u) structure, user registers
1145                            <1>                   ; and interrupt return components (for IRET)
1146 0000DC83 8705[B8030300]   <1>       xchg  eax, [u.pgdir] ; page directory of the child process
1147 0000DC89 A3[BC030300]     <1>       mov   [u.ppgdir], eax ; page directory of the parent process
1148 0000DC8E 5E               <1>       pop   esi
1149 0000DC8F 58               <1>       pop   eax   ; UPAGE (user structure page) address
1150                            <1>             ; [u.usp] = esp
1151 0000DC90 89F7             <1>       mov   edi, esi
1152 0000DC92 66C1E702         <1>       shl   di, 2
1153 0000DC96 8987[BC000300]   <1>       mov   [edi+p.upage-4], eax ; memory page for 'user' struct
1154 0000DC9C A3[B4030300]     <1>       mov   [u.upage], eax ; memory page for 'user' struct (child)
1155                            <1>       ; 28/08/2015
1156 0000DCA1 0FB605[B3030300] <1>       movzx eax, byte [u.uno] ; parent process number
1157                            <1>             ; movb u.uno,-(sp) / save parent process number
1158 0000DCA8 89C7             <1>       mov   edi, eax
1159 0000DCAA 50               <1>         push eax ; **
1160 0000DCAB 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 0000DCB1 668986[7F000300] <1>       mov    [esi+p.ttyc-1], ax ; al - set child's console tty
1165                            <1>                               ; ah - reset child's wait channel
1166 0000DCB8 89F0             <1>       mov   eax, esi
1167 0000DCBA A2[B3030300]     <1>       mov   [u.uno], al ; child process number
1168                            <1>             ;movb r1,u.uno / set child process number to r1
1169 0000DCBF 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 0000DCC5 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 0000DCC6 BB[54030300]     <1>       mov   ebx, runq+2 ; normal run queue ; 02/01/2017
1182 0000DCCB E818350000       <1>       call  putlu
1183                            <1>             ; jsr r0,putlu / put child process on lowest priority
1184                            <1>                       ; / run queue
1185 0000DCD0 66D1E6           <1>       shl   si, 1
1186                            <1>             ; asl r1 / multiply r1 by 2 to get index
1187                            <1>                    ; / into p.pid table
1188 0000DCD3 66FF05[4E030300] <1>       inc   word [mpid]
1189                            <1>             ; inc mpid / increment m.pid; get a new process name
1190 0000DCDA 66A1[4E030300]   <1>       mov   ax, [mpid]
1191 0000DCE0 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 0000DCE7 5A               <1>       pop   edx  ; * return address for the child process
1195                            <1>                  ; * Retro UNIX 8086 v1 feature only !
1196 0000DCE8 5B               <1>       pop   ebx  ; **
1197                            <1>       ;mov  ebx, [esp] ; ** parent process number
1198                            <1>             ; movb (sp),r2 / put parent process number in r2
1199 0000DCE9 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 0000DCEC 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 0000DCF3 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 0000DCFA 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 0000DCFF 8B2D[5C030300]   <1>       mov   ebp, [u.sp] ; points to return address (EIP for IRET)
1212 0000DD05 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 0000DD08 68[ECD80000]     <1>       push  sysret ; ***
1221 0000DD0D 8925[60030300]   <1>       mov   [u.usp], esp ; points to 'sysret' address (***)
1222                            <1>                          ; (for child process)
1223 0000DD13 31C0             <1>       xor   eax, eax
1224 0000DD15 66A3[94030300]   <1>       mov   [u.ttyp], ax ; 0
1225                            <1>       ;
1226 0000DD1B E858340000       <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 0000DD20 58                        <1>         pop    eax ; ***
1233 0000DD21 66D1E3                    <1>         shl    bx, 1
1234 0000DD24 8B83[BC000300]            <1>         mov     eax, [ebx+p.upage-4] ; UPAGE address ; 14/05/2015
1235 0000DD2A E881340000                <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 0000DD2F 0FB705[4E030300]          <1>           movzx   eax, word [mpid]
1239 0000DD36 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 0000DD3B 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 0000DD3D 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 0000DD43 08C0                     <1>         or     al, al
1256 0000DD45 740D                     <1>         jz     short sysfork_5
1257                                    <1>             ; beq 2f / file has not been opened by parent,
1258                                    <1>                      ; / so branch
1259 0000DD47 B40A                     <1>         mov    ah, 10 ; Retro UNIX 386 v1 fsp structure size = 10 bytes
1260 0000DD49 F6E4                     <1>         mul    ah
1261                                    <1>         ;movzx ebx, ax
1262 0000DD4B 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 0000DD4E 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 0000DD54 46                       <1>           inc     esi
1273                                    <1>             ; inc r1 / get next open file
1274 0000DD55 6683FE0A                 <1>           cmp     si, 10
1275                                    <1>             ; cmp r1,$10. / 10. files is the maximum number which
1276                                    <1>                     ; / can be opened
1277 0000DD59 72E2                     <1>         jb     short sysfork_4
1278                                    <1>             ; blt 1b / check next entry
1279 0000DD5B 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>         ;
```

```
                        <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>       ; TRDOS 386 (10/10/2016)
                        <1>       ;
                        <1>         ; INPUT ->
                        <1>         ;       CL = File Attributes
                        <1>       ;            bit 0 (1) - Read only file (R)
                        <1>       ;              bit 1 (1) - Hidden file (H)
                        <1>         ;              bit 2 (1) - System file (R)
                        <1>       ;              bit 3 (1) - Volume label/name (V)
                        <1>         ;              bit 4 (1) - Subdirectory (D)
                        <1>       ;            bit 5 (1) - File has been archived (A)
                        <1>         ;           EBX = Pointer to filename (ASCIIZ) -path-
                        <1>       ;
                        <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)
                        <1>       ;
                        <1>       ; Note: If the file is existing and it has not any one
                        <1>       ;     of S,H,R,V,D attributes, it will be truncated
                        <1>       ;     to zero length; otherwise, access error will be
                        <1>       ;     returned.
                        <1>
                        <1> sysmkdir_0:
0000DD60 F6C108         <1>       test   cl, 08h ; Volume name
0000DD63 740A           <1>       jz     short syscreat_0
                        <1>
                        <1>       ; Volume name or long name creation
                        <1>       ; is not permitted (in TRDOS 386)!
0000DD65 B80B000000     <1>       mov    eax, ERR_FILE_ACCESS  ; 11 ; 'permission denied !'
0000DD6A E926020000     <1>         jmp  sysopen_dev_err
                        <1>
                        <1> syscreat_0:
                        <1>         ;mov [u.namep], ebx
0000DD6F 51             <1>       push   ecx
0000DD70 89DE           <1>       mov    esi, ebx
                        <1>       ; file name is forced, change directory as temporary
                        <1>       ;mov   ax, 1
                        <1>       ;mov   [FFF_Valid], ah ; 0 ; reset ; 17/10/2016
                        <1>       ;call  set_working_path
0000DD72 E8C9490000     <1>       call   set_working_path_x ; 17/10/2016
0000DD77 0F82D7000000   <1>       jc     syscreat_err
                        <1>
                        <1>       ; 16/10/2016
0000DD7D 803D[478E0100]00 <1>     cmp    byte [SWP_inv_fname], 0
0000DD84 776C           <1>       ja     short  syscreat_inv_fname ; invalid file name !
                        <1>
                        <1>       ; Here, we have a valid path and also a valid file name
                        <1>       ; (Working dir has been changed if the path
                        <1>       ;  -file name string- had contained a dir name.)
                        <1>
0000DD86 6631C0         <1>       xor    ax, ax
                        <1>       ;mov   esi, FindFile_Name
0000DD89 E8E3B6FFFF     <1>       call   find_first_file
0000DD8E 59             <1>       pop    ecx
                        <1>             ; ESI = Directory Entry (FindFile_DirEntry) Location
                        <1>             ; EDI = Directory Buffer Directory Entry Location
                        <1>             ; EAX = File Size
                        <1>             ;  BL = Attributes of The File/Directory
                        <1>             ;  BH = Long Name Yes/No Status (>0 is YES)
                        <1>             ;  DX > 0 : Ambiguous filename chars are used
0000DD8F 7269           <1>       jc     short syscreat_1 ; file not found (the good!)
                        <1>                           ; or another error (the bad')
                        <1>
                        <1>       ; (& the uggly!) truncate file to zero length before open
                        <1>
                        <1>       ;'*' and '?' already checked at 'set_working_path' stage
                        <1>       ;and   dx, dx
                        <1>       ;jnz   short sysmkdir_err ; permission denied
                        <1>                             ; invalid filename chars
                        <1>
                        <1>       ;test  cl, 10h ; subdirectory ?
                        <1>       ;jnz   short sysmkdir_err
                        <1>
                        <1>       ; BL = File Attributes:
                        <1>       ;            bit 0 (1) - Read only file (R)
                        <1>       ;              bit 1 (1) - Hidden file (H)
                        <1>         ;              bit 2 (1) - System file (R)
                        <1>       ;              bit 3 (1) - Volume label/name (V)
                        <1>         ;              bit 4 (1) - Subdirectory (D)
                        <1>       ;            bit 5 (1) - File has been archived
                        <1>
                        <1>       ; * existing directory must not be truncated
                        <1>       ;   (we don't know it is empty or not, at this stage)
                        <1>       ; * existing volume name (or a long name) can not be
                        <1>       ;   re-created or truncated by 'syscreat'
                        <1>       ; * A file with S, H, R attributes must not be truncated
                        <1>       ;   (change attributes to normal, if you need truncate it)
                        <1>
0000DD91 F6C31F         <1>       test   bl, 00011111b  ; check attributes of existing file
0000DD94 754E           <1>       jnz    short sysmkdir_err
                        <1>
                        <1>       ;; normal file, OK to continue...
                        <1>
                        <1>       ; ESI = FindFile_DirEntry
0000DD96 668B4614       <1>       mov    ax, [esi+DirEntry_FstClusHI] ; 20
0000DD9A C1E010         <1>       shl    eax, 16 ; 13/11/2017
0000DD9D 668B461A       <1>       mov    ax, [esi+DirEntry_FstClusLO] ; 26
```

```
1438                                    <1>       ; EAX = First cluster to be truncated/unlinked
1439 0000DDA1 57                       <1>       push   edi
1440 0000DDA2 51                       <1>       push   ecx
1441 0000DDA3 BE00010900               <1>       mov    esi, Logical_DOSDisks
1442 0000DDA8 29C9                     <1>       sub    ecx, ecx
1443 0000DDAA 8A2D[56820100]           <1>       mov    ch, [Current_Drv]
1444 0000DDB0 01CE                     <1>       add    esi, ecx
1445                                    <1>       ; ESI = Logical dos drive description table address
1446 0000DDB2 E8C9F7FFFF               <1>       call   truncate_cluster_chain
1447 0000DDB7 59                       <1>       pop    ecx
1448 0000DDB8 5F                       <1>       pop    edi
1449 0000DDB9 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 0000DDBB 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 0000DDC0 66894716                 <1>       mov    [edi+DirEntry_WrtTime], ax
1458 0000DDC4 66895718                 <1>       mov    [edi+DirEntry_WrtDate], dx
1459 0000DDC8 66895712                 <1>       mov    [edi+DirEntry_LastAccDate], dx
1460 0000DDCC 31C0                     <1>       xor    eax, eax ; file size = 0
1461 0000DDCE 89471C                   <1>       mov    [edi+DirEntry_FileSize], eax ; 0
1462 0000DDD1 C605[7C890100]02         <1>       mov    byte [DirBuff_ValidData], 2 ; data changed sign
1463 0000DDD8 BE[488B0100]             <1>       mov    esi, FindFile_DirEntry
1464 0000DDDD B201                     <1>       mov    dl, 1 ; open file for writing
1465 0000DDDF E9AA000000               <1>       jmp    sysopen_2
1466                                    <1>
1467                                    <1> sysmkdir_err:
1468                                    <1>       ; 1 = write, 2 = read & write, >2 = invalid
1469 0000DDE4 B80B000000               <1>         mov  eax, ERR_FILE_ACCESS  ; 11 ; 'permission denied !'
1470 0000DDE9 EB73                     <1>         jmp  short sysopen_err
1471                                    <1>
1472                                    <1> syscreate_truncate_err:
1473 0000DDEB B812000000               <1>       mov    eax, ERR_DRV_WRITE ; 18 ; 'disk write error !'
1474 0000DDF0 EB6C                     <1>         jmp  short sysopen_err
1475                                    <1>
1476                                    <1> syscreat_inv_fname:  ; invalid file name chars
1477                                    <1>       ; 16/10/2016
1478 0000DDF2 B81A000000               <1>       mov    eax, ERR_INV_FILE_NAME  ; 26 ; invalid file name chars
1479 0000DDF7 59                       <1>       pop    ecx
1480 0000DDF8 EB64                     <1>       jmp    sysopen_err
1481                                    <1>
1482                                    <1> syscreat_1:
1483                                    <1>       ; Error code in EAX
1484 0000DDFA 3C02                     <1>         cmp  al, 02h ; 'File not found' error
1485 0000DDFC 7560                     <1>         jne  sysopen_err
1486                                    <1>
1487 0000DDFE F6C110                   <1>       test   cl, 10h ; Directory
1488 0000DE01 0F852C020000             <1>       jnz    sysmkdir_2
1489                                    <1>
1490                                    <1> syscreat_2:
1491 0000DE07 BE[388B0100]             <1>       mov    esi, FindFile_Name
1492                                    <1>         ;xor edx, edx
1493 0000DE0C 31C0                     <1>         xor  eax, eax ; File Size  = 0
1494 0000DE0E 31DB                     <1>       xor    ebx, ebx
1495 0000DE10 4B                       <1>       dec    ebx ; FFFFFFFFh -> create empty file
1496                                    <1>                    ;              (only for FAT fs)
1497                                    <1>       ; CL = File Attributes
1498 0000DE11 E8F8EBFFFF               <1>       call   create_file
1499 0000DE16 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
1513                                    <1>       ; been updated, yet!
1514                                    <1>       ; (Note: Directory and FAT buffers have been updated...)
1515                                    <1>
1516 0000DE18 E824DDFFFF               <1>       call   update_parent_dir_lmdt ; now, it is OK too!
1517                                    <1>
1518                                    <1>       ; 25/10/2016
1519 0000DE1D 66B80018                 <1>       mov    ax, 1800h
1520 0000DE21 BE[388B0100]             <1>       mov    esi, FindFile_Name
1521 0000DE26 E846B6FFFF               <1>       call   find_first_file
1522 0000DE2B 7231                     <1>       jc     short sysopen_err
1523                                    <1>
1524                                    <1>       ; Only possible error after here is
1525                                    <1>       ; "too many open files !" error.
1526                                    <1>       ;
1527                                    <1>       ; If "syscreat" will return with that error,
1528                                    <1>       ; (the file has been created but it could not be opened)
1529                                    <1>       ; the user must retry to open this file again
1530                                    <1>       ; or must close another file before using
1531                                    <1>       ; "sysopen" system call.
1532                                    <1>
1533 0000DE2D B201                     <1>       mov    dl, 1 ; open file for writing
1534                                    <1>       ; ESI = Directory Entry (FindFile_DirEntry) Location
1535                                    <1>       ; EAX = File Size (= 0)
1536 0000DE2F EB5D                     <1>       jmp    short sysopen_2
1537                                    <1>
1538                                    <1> sysopen: ;<open file>
1539                                    <1>       ; 26/10/2016
1540                                    <1>       ; 24/10/2016
1541                                    <1>       ; 17/10/2016
1542                                    <1>       ; 15/10/2016
```

```
1543                                    <1>       ; 06/10/2016, 07/10/2016, 08/10/2016
1544                                    <1>       ; 05/10/2016 (TRDOS 386 = TRDOS v2.0)
1545                                    <1>       ;          -derived from INT_21H.ASM-
1546                                    <1>       ;            ("loc_INT21h_open_file")
1547                                    <1>         ;    26/02/2011
1548                                    <1>         ;    INT 21h Function AH = 3Dh
1549                                    <1>         ;    Open File
1550                                    <1>         ;    INPUT
1551                                    <1>         ;       AL= File Access Value
1552                                    <1>       ;           0- Open for reading
1553                                    <1>       ;            1- Open for writing
1554                                    <1>         ;           2- Open for reading and writing
1555                                    <1>         ;          DS:DX= Pointer to filename (ASCIIZ)
1556                                    <1>         ;
1557                                    <1>       ; 14/05/2015 (Retro UNIX 386 v1 - Beginning)
1558                                    <1>       ; 22/05/2013 - 27/05/2013 (Retro UNIX 8086 v1)
1559                                    <1>       ;
1560                                    <1>       ; 'sysopen' opens a file in following manner:
1561                                    <1>       ;    1) The second argument in a sysopen says whether to
1562                                    <1>       ;     open the file ro read (0) or write (>0).
1563                                    <1>       ;    2) I-node of the particular file is obtained via 'namei'.
1564                                    <1>       ;    3) The file is opened by 'iopen'.
1565                                    <1>       ;    4) Next housekeeping is performed on the fsp table
1566                                    <1>       ;     and the user's open file list - u.fp.
1567                                    <1>       ;     a) u.fp and fsp are scanned for the next available slot.
1568                                    <1>       ;     b) An entry for the file is created in the fsp table.
1569                                    <1>       ;     c) The number of this entry is put on u.fp list.
1570                                    <1>       ;     d) The file descriptor index to u.fp list is pointed
1571                                    <1>       ;        to by u.r0.
1572                                    <1>       ;
1573                                    <1>       ; Calling sequence:
1574                                    <1>       ;     sysopen; name; mode
1575                                    <1>       ; Arguments:
1576                                    <1>       ;     name - file name or path name
1577                                    <1>       ;     mode - 0 to open for reading
1578                                    <1>       ;            1 to open for writing
1579                                    <1>       ; Inputs: (arguments)
1580                                    <1>       ; Outputs: *u.r0 - index to u.fp list (the file descriptor)
1581                                    <1>       ;             is put into r0's location on the stack.
1582                                    <1>       ; ..........................................................
1583                                    <1>       ;
1584                                    <1>       ; Retro UNIX 8086 v1 modification:
1585                                    <1>       ;      'sysopen' system call has two arguments; so,
1586                                    <1>       ;     * 1st argument, name is pointed to by BX register
1587                                    <1>       ;     * 2nd argument, mode is in CX register
1588                                    <1>       ;
1589                                    <1>       ;     AX register (will be restored via 'u.r0') will return
1590                                    <1>       ;     to the user with the file descriptor/number
1591                                    <1>       ;     (index to u.fp list).
1592                                    <1>       ;
1593                                    <1>       ;call  arg2
1594                                    <1>       ; * name - 'u.namep' points to address of file/path name
1595                                    <1>       ;          in the user's program segment ('u.segmnt')
1596                                    <1>       ;          with offset in BX register (as sysopen argument 1).
1597                                    <1>       ; * mode - sysopen argument 2 is in CX register
1598                                    <1>       ;          which is on top of stack.
1599                                    <1>       ;
1600                                    <1>       ; jsr r0,arg2 / get sys args into u.namep and on stack
1601                                    <1>       ;
1602                                    <1>            ; system call registers: ebx, ecx (through 'sysenter')
1603                                    <1>       ;
1604                                    <1>       ; TRDOS 386 (05/10/2016)
1605                                    <1>       ;
1606                                    <1>         ; INPUT ->
1607                                    <1>         ;      CL = File Access Value (Open Mode)
1608                                    <1>       ;           0 - Open file for reading
1609                                    <1>       ;            1 - Open file for writing
1610                                    <1>         ;            2 - Open device for reading
1611                                    <1>       ;           3 - Open device for writing
1612                                    <1>         ;          EBX = Pointer to filename/devicename (ASCIIZ)
1613                                    <1>       ; OUTPUT ->
1614                                    <1>       ;          eax = File/Device Handle/Number (index) (AL)
1615                                    <1>       ;          cf = 1 -> Error code in AL
1616                                    <1>       ;
1617                                    <1>       ; Modified Registers: EAX (at the return of system call)
1618                                    <1>       ;
1619                                    <1>
1620 0000DE31 80F901                  <1>       cmp    cl, 1 ; read file (0), write file (1)
1621 0000DE34 7614                    <1>       jna    short sysopen_0
1622                                    <1>
1623 0000DE36 80F903                  <1>       cmp    cl, 3
1624 0000DE39 0F8640010000            <1>       jna    sysopen_device
1625                                    <1>
1626                                    <1>       ; Invalid access code
1627 0000DE3F B817000000              <1>       mov    eax, ERR_INV_PARAMETER
1628 0000DE44 0F874B010000            <1>       ja     sysopen_dev_err
1629                                    <1>
1630                                    <1> sysopen_0:
1631                                    <1>       ;mov   [u.namep], ebx
1632 0000DE4A 51                      <1>       push   ecx
1633 0000DE4B 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 0000DE4D E8EE480000              <1>       call   set_working_path_x ; 17/10/2016
1639 0000DE52 731E                    <1>       jnc    short sysopen_1
1640                                    <1>
1641                                    <1> syscreat_err: ; ecx = file attributes (for 'syscreat')
1642 0000DE54 59                      <1>       pop    ecx ; open mode
1643 0000DE55 21C0                    <1>       and    eax, eax  ; 0 -> Bad Path!
1644 0000DE57 7505                    <1>       jnz    short sysopen_err
1645                                    <1>       ; eax = 0
1646 0000DE59 B80C000000              <1>       mov    eax, ERR_DIR_NOT_FOUND ; Directory not found !
1647                                    <1> sysopen_err:
```

```
1648 0000DE5E A3[64030300]        <1>       mov   [u.r0], eax
1649 0000DE63 A3[C8030300]        <1>       mov   [u.error], eax
1650 0000DE68 E8A8490000          <1>       call  reset_working_path
1651 0000DE6D E95AFAFFFF          <1>       jmp   error
1652                              <1>
1653                              <1> sysopen_1:
1654                              <1>       ;mov  esi, FindFile_Name
1655 0000DE72 66B80018            <1>         mov ax, 1800h ; Only files
1656 0000DE76 E8F6B5FFFF          <1>       call  find_first_file
1657 0000DE7B 5A                  <1>       pop   edx
1658 0000DE7C 72E0                <1>       jc    short sysopen_err ; eax = 2 (File not found !)
1659                              <1>
1660                              <1>       ; check_open_file_attr_access_code
1661                              <1>
1662 0000DE7E F6C307              <1>         test bl, 7  ; system, hidden, readonly
1663 0000DE81 740B                <1>         jz  short sysopen_2
1664                              <1>
1665 0000DE83 20D2                <1>       and   dl, dl ; 0 = read mode
1666 0000DE85 7407                <1>       jz    short sysopen_2
1667                              <1>
1668                              <1>       ; 1 = write, 2 = read & write, >2 = invalid
1669 0000DE87 B80B000000          <1>         mov eax, ERR_FILE_ACCESS  ; 11 = 'permission denied !'
1670 0000DE8C EBD0                <1>         jmp short sysopen_err
1671                              <1>
1672                              <1> sysopen_2:
1673                              <1>       ; esi = Directory Entry (FindFile_DirEntry) Location
1674 0000DE8E 89F3                <1>       mov   ebx, esi
1675 0000DE90 31F6                <1>         xor     esi, esi ; 0
1676 0000DE92 31FF                <1>         xor     edi, edi ; 0
1677                              <1> sysopen_3: ; scan the list of entries in fsp table
1678 0000DE94 80BE[6A030300]00    <1>         cmp     byte [esi+u.fp], 0
1679 0000DE9B 760F                <1>         jna     short sysopen_4 ; empty slot
1680 0000DE9D 6646                <1>         inc     si
1681 0000DE9F 6683FE0A            <1>         cmp     si, 10
1682 0000DEA3 72EF                <1>       jb    short sysopen_3
1683                              <1> toomanyf:
1684 0000DEA5 B80D000000          <1>       mov   eax, ERR_TOO_MANY_FILES ; too many open files !
1685 0000DEAA EBB2                <1>       jmp   short sysopen_err
1686                              <1>
1687                              <1> sysopen_4:
1688 0000DEAC 80BF[B6910100]00    <1>         cmp     byte [edi+OF_MODE], 0 ; Scan open files table
1689 0000DEB3 760A                <1>       jna    short sysopen_5
1690 0000DEB5 6647                <1>       inc   di
1691 0000DEB7 6683FF0A            <1>       cmp    di, OPENFILES ; max. number of open files (=10)
1692 0000DEBB 72EF                <1>       jb    short sysopen_4
1693 0000DEBD EBE6                <1>       jmp   short toomanyf
1694                              <1>
1695                              <1> sysopen_5:
1696 0000DEBF FEC2                <1>       inc   dl
1697 0000DEC1 8897[B6910100]      <1>         mov     [edi+OF_MODE], dl
1698 0000DEC7 8A15[F68A0100]      <1>       mov   dl, [FindFile_Drv]
1699 0000DECD 8897[AC910100]      <1>         mov     [edi+OF_DRIVE], dl ; Logical DOS drive number
1700 0000DED3 66C1E702            <1>       shl   di, 2 ; *4 (dword offset)
1701                              <1>
1702 0000DED7 8987[FC910100]      <1>       mov   [edi+OF_SIZE], eax ; File size in bytes
1703                              <1>
1704 0000DEDD 668B4314            <1>         mov ax, [ebx+DirEntry_FstClusHI]
1705 0000DEE1 C1E010              <1>       shl   eax, 16
1706 0000DEE4 668B431A            <1>       mov   ax, [ebx+DirEntry_FstClusLO]
1707 0000DEE8 8987[84910100]      <1>       mov    [edi+OF_FCLUSTER], eax ; First cluster
1708 0000DEEE 8987[9C920100]      <1>       mov    [edi+OF_CCLUSTER], eax ; Current cluster
1709                              <1>
1710 0000DEF4 31DB                <1>         xor ebx, ebx
1711 0000DEF6 899F[D4910100]      <1>         mov     [edi+OF_POINTER], ebx ; offset pointer (0)
1712 0000DEFC 899F[C4920100]      <1>         mov     [edi+OF_CCINDEX], ebx ; cluster index (0)
1713                              <1>
1714 0000DF02 A1[688B0100]        <1>       mov   eax, [FindFile_DirFirstCluster]
1715 0000DF07 8987[24920100]      <1>       mov   [edi+OF_DIRFCLUSTER], eax
1716                              <1>
1717 0000DF0D A1[6C8B0100]        <1>       mov   eax, [FindFile_DirCluster]
1718 0000DF12 8987[4C920100]      <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 0000DF18 88D7                <1>       mov   bh, dl
1724 0000DF1A 81C300010900        <1>         add ebx, Logical_DOSDisks
1725 0000DF20 8A4303              <1>         mov al, [ebx+LD_FATType]
1726 0000DF23 3C01                <1>         cmp al, 1
1727 0000DF25 7209                <1>         jb  short sysopen_6_fs
1728 0000DF27 3C02                <1>         cmp al, 2
1729 0000DF29 770A                <1>         ja  short sysopen_6_fat32
1730                              <1> sysopen_6_fat:
1731 0000DF2B 8B432D              <1>         mov eax, [ebx+LD_BPB+VolumeID]
1732 0000DF2E EB08                <1>         jmp short sysopen_7
1733                              <1> sysopen_6_fs:
1734 0000DF30 8B4328              <1>         mov eax, [ebx+LD_FS_VolumeSerial]
1735 0000DF33 EB03                <1>         jmp short sysopen_7
1736                              <1> sysopen_6_fat32:
1737 0000DF35 8B4349              <1>         mov eax, [ebx+LD_BPB+FAT32_VolID]
1738                              <1> sysopen_7:
1739 0000DF38 A3[4C820100]        <1>         mov [Current_VolSerial], eax
1740                              <1>
1741 0000DF3D 8987[74920100]      <1>       mov   [edi+OF_VOLUMEID], eax
1742                              <1>
1743                              <1>       ; 24/10/2016
1744 0000DF43 66D1EF              <1>       shr   di, 1 ; 4/2, word offset
1745 0000DF46 668B1D[708B0100]    <1>       mov   bx, [FindFile_DirEntryNumber]
1746 0000DF4D 66899F[EC920100]    <1>       mov   [edi+OF_DIRENTRY], bx
1747                              <1>
1748 0000DF54 31D2                <1>       xor   edx, edx
1749                              <1>       ;shr  di, 2 ; /4 (byte offset)
1750 0000DF56 66D1EF              <1>       shr   di, 1 ; 2/2, byte offset
1751 0000DF59 8897[CA910100]      <1>       mov   byte [edi+OF_OPENCOUNT], dl ; 0
1752 0000DF5F 8897[C0910100]      <1>       mov   byte [edi+OF_STATUS], dl ; 0
```

```
1753                                    <1>
1754 0000DF65 89FB                     <1>       mov    ebx, edi
1755 0000DF67 FEC3                     <1>       inc    bl
1756                                    <1>
1757 0000DF69 889E[6A030300]            <1>         mov     [esi+u.fp], bl ; Open File Entry Number
1758 0000DF6F 8935[64030300]            <1>       mov     [u.r0], esi ; move index to u.fp list
1759                                    <1>                     ; into eax on stack
1760                                    <1>
1761 0000DF75 E89B480000                <1>         call       reset_working_path
1762                                    <1>
1763 0000DF7A 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 0000DF7F 51                       <1>       push   ecx ; open mode
1786 0000DF80 89E5                     <1>       mov    ebp, esp
1787 0000DF82 B910000000               <1>       mov    ecx, 16 ; transfer length = 16 bytes
1788 0000DF87 29CC                     <1>       sub    esp, ecx
1789 0000DF89 89E7                     <1>       mov    edi, esp ; destination address
1790 0000DF8B 89DE                     <1>       mov    esi, ebx ; dev name in user's memory space
1791 0000DF8D E87C330000               <1>       call   transfer_from_user_buffer
1792 0000DF92 7310                     <1>       jnc    short sysopen_dev_0
1793                                    <1>       ; eax = ERR_OUT_OF_MEMORY = 4 = ERR_MINOR_IM
1794 0000DF94 59                       <1>       pop    ecx
1795                                    <1> sysopen_dev_err:
1796 0000DF95 A3[64030300]             <1>       mov    [u.r0], eax
1797 0000DF9A A3[C8030300]             <1>       mov    [u.error], eax
1798 0000DF9F E928F9FFFF               <1>       jmp    error
1799                                    <1> sysopen_dev_0:
1800 0000DFA4 89FE                     <1>       mov    esi, edi ; Device name addr (max. 16 bytes, ASCIIZ)
1801                                    <1>                     ; for example: "tty, TTY, /dev/tty"
1802 0000DFA6 E8104B0000               <1>       call   get_device_number
1803 0000DFAB 89EC                     <1>       mov    esp, ebp
1804 0000DFAD 59                       <1>       pop    ecx
1805 0000DFAE 7307                     <1>       jnc    short sysopen_dev_1
1806 0000DFB0 B818000000               <1>       mov    eax, ERR_INV_DEV_NAME ; 24 ; 'invalid device name !'
1807 0000DFB5 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 0000DFB7 31DB                     <1>         xor     ebx, ebx ; 0
1812                                    <1> sysopen_dev_2: ; scan the list of entries
1813 0000DFB9 389B[6A030300]           <1>         cmp     [ebx+u.fp], bl ; 0
1814 0000DFBF 760E                     <1>         jna     short sysopen_dev_3 ; empty slot
1815 0000DFC1 FEC3                     <1>         inc     bl
1816 0000DFC3 80FB0A                   <1>         cmp     bl, 10
1817 0000DFC6 72F1                     <1>       jb     short sysopen_dev_2
1818                                    <1>       ;
1819 0000DFC8 B80D000000               <1>       mov    eax, ERR_TOO_MANY_FILES ; too many open files !
1820 0000DFCD EBC6                     <1>       jmp    short sysopen_dev_err
1821                                    <1> sysopen_dev_3:
1822 0000DFCF 891D[64030300]           <1>       mov    [u.r0], ebx ; File/Device index/handle/descriptor
1823                                    <1>       ; eax = device number (entry offset)
1824 0000DFD5 8AA8[488F0100]           <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 0000DFDB F6C501                   <1>         test   ch, 1 ; accessable by normal users (except root)
1833 0000DFDE 7510                     <1>         jnz    short sysopen_dev_4 ; yes, permission has been given
1834 0000DFE0 803D[B0030300]00         <1>         cmp    byte [u.uid], 0 ; root?
1835 0000DFE7 7607                     <1>         jna    short sysopen_dev_4 ; superuser can open all devices
1836                                    <1> sysopen_dev_perm_err:
1837 0000DFE9 B80B000000               <1>       mov    eax, ERR_DEV_ACCESS  ; 11 = 'permission denied !'
1838 0000DFEE EBA5                     <1>       jmp    short sysopen_dev_err
1839                                    <1> sysopen_dev_4:
1840 0000DFF0 D0ED                     <1>       shr    ch, 1 ; result: 1 = read, 2 = write, 3 = r & w
1841 0000DFF2 FEC9                     <1>       dec    cl  ; result: 1 = read, 2 = write
1842 0000DFF4 84E9                     <1>       test   cl, ch
1843 0000DFF6 74F1                     <1>       jz     short sysopen_dev_perm_err
1844                                    <1>
1845 0000DFF8 D0E5                     <1>       shl    ch, 1 ; bit 0 = 0
1846                                    <1>       ; eax = device number (entry offset)
1847 0000DFFA E8D84B0000               <1>       call   device_open
1848 0000DFFF 72E8                     <1>       jc     short sysopen_dev_perm_err
1849                                    <1>
1850                                    <1>       ; eax = device number (entry offset)
1851 0000E001 0C80                     <1>       or     al, 80h ; set device bit (set bit 7 to 1)
1852 0000E003 8B1D[64030300]           <1>       mov    ebx, [u.r0]
1853 0000E009 8883[6A030300]           <1>       mov    [ebx+u.fp], al      ; bit 7 (=1) points to device
1854                                    <1>
1855 0000E00F 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 0000E014 6621C9                  <1>       and   cx, cx ; if cx = 0 -> create a normal subdir
1918 0000E017 7413                    <1>       jz    short sysmkdir_1
1919                                  <1>
1920 0000E019 F6C110                  <1>       test  cl, 10h ; if dir flags set, also use other flags
1921 0000E01C 0F853EFDFFFF            <1>       jnz   sysmkdir_0 ; jump to head of 'syscreat'
1922                                  <1>
1923                                  <1>       ; CX has wrong flags
1924 0000E022 B817000000              <1>       mov   eax, ERR_INV_FLAGS
1925 0000E027 E969FFFFFF              <1>       jmp   sysopen_dev_err
1926                                  <1>
1927                                  <1> sysmkdir_1:
1928 0000E02C B110                    <1>       mov   cl, 10h ; set subdir flag and reset other flags
1929 0000E02E 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 0000E033 BE[388B0100]            <1>       mov   esi, FindFile_Name
1934 0000E038 E804D7FFFF              <1>       call  make_sub_directory
1935 0000E03D 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 0000E043 A3[64030300]            <1>       mov   [u.r0], eax ; New sub dir's first cluster
1950                                  <1>
1951 0000E048 E8C8470000              <1>         call      reset_working_path
1952                                  <1>
1953 0000E04D 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 0000E052 89D8                     <1>       mov    eax, ebx
1990 0000E054 31DB                     <1>       xor    ebx, ebx
1991 0000E056 891D[64030300]           <1>       mov    [u.r0], ebx ; 0  ; return value of EAX
1992 0000E05C E83C270000               <1>       call   fclose
1993 0000E061 0F8385F8FFFF             <1>       jnc    sysret
1994 0000E067 B80A000000               <1>       mov    eax, ERR_FILE_NOT_OPEN ; file not open !
1995 0000E06C A3[C8030300]             <1>       mov    [u.error], eax ;
1996 0000E071 A3[64030300]             <1>       mov    [u.r0], eax ; ! invalid handle !
1997 0000E076 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,
2038                                    <1>       ;     * 1st argument, file descriptor is in BX register
2039                                    <1>       ;     * 2nd argument, buffer address/offset in CX register
2040                                    <1>       ;     * 3rd argument, number of bytes is in DX register
2041                                    <1>       ;
2042                                    <1>       ;     AX register (will be restored via 'u.r0') will return
2043                                    <1>       ;     to the user with number of bytes read.
2044                                    <1>       ;
2045                                    <1>       ; TRDOS 386 (05/10/2016)
2046                                    <1>       ;
2047                                    <1>         ; INPUT ->
2048                                    <1>         ;      EBX = File handle (descriptor/index)
2049                                    <1>       ;        ECX = Buffer address
2050                                    <1>         ;          EDX = Number of bytes
2051                                    <1>       ; OUTPUT ->
2052                                    <1>       ;         EAX = Number of bytes have been read
2053                                    <1>       ;         cf = 1 -> Error code in AL
2054                                    <1>       ;
2055                                    <1>       ; Modified Registers: EAX (at the return of system call)
2056                                    <1>       ;
2057                                    <1>
2058                                    <1>       ; EBX = File descriptor
2059 0000E07B E86B270000               <1>       call   getf1
2060 0000E080 7277                     <1>       jc     short device_read ; read data from device
2061                                    <1>       ; EAX = First cluster of the file
2062                                    <1>
2063 0000E082 E83F000000               <1>       call   rw1
2064 0000E087 730A                     <1>       jnc    short sysread_0
2065                                    <1>
2066 0000E089 A3[64030300]             <1>       mov    [u.r0], eax ; error code
2067 0000E08E E939F8FFFF               <1>       jmp    error
```

```
2068                                    <1>
2069                                    <1> sysread_0:
2070 0000E093 E8622D0000                <1>         call   readi
2071 0000E098 EB1D                      <1>         jmp    short rw0
2072                                    <1>
2073                                    <1> syswrite: ; < write to file >
2074                                    <1>         ; 23/10/2016
2075                                    <1>         ; 11/10/2016 (TRDOS 386 = TRDOS v2.0)
2076                                    <1>         ;           -derived from INT_21H.ASM-
2077                                    <1>         ;             ("loc_INT21h_write_file")
2078                                    <1>           ;    13/03/2011 (05/03/2011)
2079                                    <1>           ;    INT 21h Function AH = 40h
2080                                    <1>           ;    Write to a File
2081                                    <1>           ;    INPUT
2082                                    <1>         ;          BX = File Handle
2083                                    <1>           ;       CX = Number of bytes to write
2084                                    <1>           ;           DS:DX= Buffer address
2085                                    <1>           ;
2086                                    <1>         ; Note: TRDOS 386 'sysrwrite' has been derived from
2087                                    <1>         ;     Retro UNIX 386 v1 'syswrite', except a few
2088                                    <1>         ;     code modifications.
2089                                    <1>         ;
2090                                    <1>
2091                                    <1>         ; 13/05/2015 (Retro UNIX 386 v1)
2092                                    <1>         ; 11/05/2015 (Retro UNIX 386 v1 - Beginning)
2093                                    <1>         ; 23/05/2013 (Retro UNIX 8086 v1)
2094                                    <1>         ;
2095                                    <1>         ; 'syswrite' is given a buffer to write onto an output file
2096                                    <1>         ; and the number of characters to write. If finds the file
2097                                    <1>         ; from the file descriptor located in *u.r0 (r0). This file
2098                                    <1>         ; descriptor is returned from a successful open or create call
2099                                    <1>         ; (sysopen or syscreat). The i-number of file is obtained via
2100                                    <1>         ; 'rw1' and buffer is written on the output file via 'write'.
2101                                    <1>         ;
2102                                    <1>         ; Calling sequence:
2103                                    <1>         ;     syswrite; buffer; nchars
2104                                    <1>         ; Arguments:
2105                                    <1>         ;     buffer - location of contiguous bytes to be writtten.
2106                                    <1>         ;     nchars - number of characters to be written.
2107                                    <1>         ; Inputs: *u.r0 - file descriptor (& arguments)
2108                                    <1>         ; Outputs: *u.r0 - number of bytes written.
2109                                    <1>         ; ..........................................................
2110                                    <1>         ;
2111                                    <1>         ; Retro UNIX 8086 v1 modification:
2112                                    <1>         ;       'syswrite' system call has three arguments; so,
2113                                    <1>         ;     * 1st argument, file descriptor is in BX register
2114                                    <1>         ;     * 2nd argument, buffer address/offset in CX register
2115                                    <1>         ;     * 3rd argument, number of bytes is in DX register
2116                                    <1>         ;
2117                                    <1>         ;     AX register (will be restored via 'u.r0') will return
2118                                    <1>         ;     to the user with number of bytes written.
2119                                    <1>         ;
2120                                    <1>         ; INPUT ->
2121                                    <1>           ;      EBX = File handle (descriptor/index)
2122                                    <1>         ;         ECX = Buffer address
2123                                    <1>           ;          EDX = Number of bytes
2124                                    <1>         ; OUTPUT ->
2125                                    <1>         ;          EAX = Number of bytes have been written
2126                                    <1>         ;          cf = 1 -> Error code in AL
2127                                    <1>         ;
2128                                    <1>         ; Modified Registers: EAX (at the return of system call)
2129                                    <1>         ;
2130                                    <1>
2131                                    <1>         ; EBX = File descriptor
2132 0000E09A E84C270000                <1>         call   getf1
2133 0000E09F 7274                      <1>         jc     short device_write ; write data to device
2134                                    <1>         ; EAX = First cluster of the file
2135                                    <1>         ; EBX = File number  (Open file number) ; 23/10/2016
2136                                    <1>
2137 0000E0A1 E820000000                <1>         call   rw1
2138 0000E0A6 730A                      <1>         jnc    short syswrite_0
2139 0000E0A8 A3[64030300]              <1>         mov    [u.r0], eax ; error code
2140 0000E0AD E91AF8FFFF                <1>         jmp    error
2141                                    <1>
2142                                    <1> syswrite_0:
2143 0000E0B2 E86F340000                <1>         call   writei
2144                                    <1> rw0: ; 1:
2145 0000E0B7 A1[8C030300]              <1>           mov  eax, [u.nread]
2146 0000E0BC A3[64030300]              <1>         mov    [u.r0], eax
2147 0000E0C1 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 0000E0C6 83F802                    <1>         cmp    eax, 2
2162 0000E0C9 7217                      <1>         jb     short rw2
2163                                    <1>         ;
2164 0000E0CB 890D[84030300]            <1>         mov    [u.base], ecx       ; buffer address/offset
2165                                    <1>                                ;(in the user's virtual memory space)
2166 0000E0D1 8915[88030300]            <1>         mov    [u.count], edx
2167                                    <1>
2168 0000E0D7 C705[C8030300]0000-       <1>           mov     dword [u.error], 0 ; reset the last error code
2168 0000E0DF 0000                      <1>
2169 0000E0E1 C3                        <1>         retn
2170                                    <1>
2171                                    <1> rw2:
```

```
2172 0000E0E2 B80A000000          <1>       mov   eax, ERR_FILE_NOT_OPEN ; file not open !
2173 0000E0E7 A3[C8030300]        <1>       mov   dword [u.error], eax
2174 0000E0EC C3                  <1>       retn
2175                              <1> rw3:
2176 0000E0ED B80B000000          <1>       mov   eax, ERR_FILE_ACCESS ; permission denied !
2177 0000E0F2 A3[C8030300]        <1>       mov   dword [u.error], eax
2178 0000E0F7 F9                  <1>       stc
2179 0000E0F8 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 0000E0F9 F6C101              <1>       test  cl, 1 ; 1 = read, 2 = write, 3 = read&write
2188 0000E0FC 74EF                <1>       jz    short rw3
2189                              <1>
2190 0000E0FE 89C3                <1>       mov   ebx, eax
2191 0000E100 66C1E302            <1>       shl   bx, 2 ; *4
2192                              <1>
2193 0000E104 F6C580              <1>       test  ch, 80h ; bit 7, installable device driver flag
2194 0000E107 7406                <1>       jz    short d_read_2 ; Kernel device
2195                              <1>       ; installable device
2196                              <1> d_read_1:
2197 0000E109 FFA3[048F0100]      <1>         jmp dword [ebx+IDEV_RADDR-4]
2198                              <1> d_read_2:
2199 0000E10F FFA3[A0400100]      <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 0000E115 F6C102              <1>       test  cl, 2 ; 1 = read, 2 = write, 3 = read&write
2208 0000E118 74D3                <1>       jz    short rw3
2209                              <1>
2210 0000E11A 89C3                <1>       mov   ebx, eax
2211 0000E11C 66C1E302            <1>       shl   bx, 2 ; *4
2212                              <1>
2213 0000E120 F6C580              <1>       test  ch, 80h ; bit 7, installable device driver flag
2214 0000E123 7406                <1>       jz    short d_write_2 ; Kernel device
2215                              <1>       ; installable device
2216                              <1> d_write_1:
2217 0000E125 FFA3[248F0100]      <1>         jmp dword [ebx+IDEV_WADDR-4]
2218                              <1> d_write_2:
2219 0000E12B FFA3[F0400100]      <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
2247                              <1>       ;       will not be changed by another process within
2248                              <1>       ;       run time sequence/schedule, if running process
2249                              <1>       ;       will not 'release' itself. Only 'wakeup' procedure
2250                              <1>       ;       for waiting processes and programmed timer events
2251                              <1>       ;       for other processes can change running process
2252                              <1>       ;       while multi tasking is disabled.) ** 23/05/2016 **
2253                              <1>
2254 0000E131 803D[B0030300]00    <1>       cmp   byte [u.uid], 0 ; root ?
2255                              <1>       ;ja   error
2256 0000E138 0F87D3F8FFFF        <1>       ja    badsys ; 14/05/2015
2257                              <1>       ;
2258 0000E13E FA                  <1>       cli
2259 0000E13F 881D[228E0100]      <1>       mov   [multi_tasking], bl ; 0 to disable, >0 to enable
2260 0000E145 E9A2F7FFFF          <1>       jmp   sysret
2261                              <1>
2262                              <1> systimer:
2263                              <1>       ; 02/01/2017
2264                              <1>       ; 21/12/2016
2265                              <1>       ; 19/12/2016
2266                              <1>       ; 10/12/2016 (callback)
2267                              <1>       ; 10/06/2016
2268                              <1>       ; 07/06/2016
2269                              <1>       ; 06/06/2016
2270                              <1>       ; 21/05/2016
2271                              <1>       ; 19/05/2016
2272                              <1>       ; 18/05/2016 - TRDOS 386 (TRDOS v2.0)
2273                              <1>       ; (TRDOS 386 feature only!)
2274                              <1>       ;
2275                              <1>       ; (start or stop timer event(s))
2276                              <1>       ;
```

```
2277                                 <1>       ; INPUT ->
2278                                 <1>       ;     BL = Signal return byte (response byte)
2279                                 <1>       ;          (Any requested value between 0 and 255)
2280                                 <1>       ;          (Kernel will put it at the requested address)
2281                                 <1>       ;     BH = Time count unit
2282                                 <1>       ;          0 = Stop timer event
2283                                 <1>       ;          1 = 18.2 ticks per second
2284                                 <1>       ;          2 = 10 milliseconds
2285                                 <1>       ;          3 = 1 second (for real time clock interrupt)
2286                                 <1>       ;          4 = time/tick count in current time count unit
2287                                 <1>       ;         // 10/12/2016
2288                                 <1>       ;         80h = Stop timer event (callback method)
2289                                 <1>       ;         81h = 18.2 ticks per second, callback method
2290                                 <1>       ;         82h = 10 milliseconds, callback method
2291                                 <1>       ;         83h = 1 second (for RTC int), callback method
2292                                 <1>       ;         84h = current time count unit, callback method
2293                                 <1>       ;
2294                                 <1>       ;         Note: Only 03h or 83h will set real time clock
2295                                 <1>       ;              (RTC) events (Others are for PIT events)!
2296                                 <1>       ;
2297                                 <1>       ;     NOTE: If callback (user service) method is used,
2298                                 <1>       ;         EDX will point to the return address (of service
2299                                 <1>       ;         procedure) in user's space instead of signal
2300                                 <1>       ;         response byte address. (TRDOS 386 kernel will
2301                                 <1>       ;         direct the cpu to that address -in user's space-
2302                                 <1>       ;         at the return of system call or interrupt
2303                                 <1>       ;         just after the adjusted count/time is elapsed.)
2304                                 <1>       ;         User's sevice routine must be ended with a
2305                                 <1>       ;         'iret'. Normal return addresses from system
2306                                 <1>       ;         calls or and interrupts will be kept same except
2307                                 <1>       ;         the timer returns.
2308                                 <1>       ;
2309                                 <1>       ;     BH = 0 -> Stop timer event
2310                                 <1>       ;     BL = Timer event number (1 to 255) if BH = 0
2311                                 <1>       ;          If BL = 0, all timer events (which are belongs
2312                                 <1>       ;           to running process) will be stopped
2313                                 <1>       ;     ECX = Time/Tick count (depending on time count unit)
2314                                 <1>       ;     EDX = Signal return (Response) byte address
2315                                 <1>       ;           (virtual address in user's memory space)
2316                                 <1>       ; OUTPUT ->
2317                                 <1>       ;     AL = Timer event number   (1 to 255) (max. value = 16)
2318                                 <1>       ;     IF BH Input = 0 & CF = 0 & AL = 0 ->
2319                                 <1>       ;          timer event(s) has/have been stopped/finished
2320                                 <1>       ;     CF = 1 & AL = 0 -> no timer setting space to set
2321                                 <1>       ;     CF = 1 & AL > 0 -> timer count unit is not usable
2322                                 <1>       ;
2323                                 <1>       ;     NOTE: To modify a time count for a user function,
2324                                 <1>       ;           at first, current timer event must be stopped
2325                                 <1>       ;           then a new timer event (which is related with
2326                                 <1>       ;           same user function) must be started.
2327                                 <1>       ;
2328                                 <1>       ;           Signal return (response) byte may be used for
2329                                 <1>       ;           several purposes. Kernel will put this value
2330                                 <1>       ;           to requested address during timer interrupt,
2331                                 <1>       ;           program/user can check this value to understand
2332                                 <1>       ;           which event has been occurred and what is changed.
2333                                 <1>       ;           (Multi timer events can share same signal address)
2334                                 <1>       ;
2335                                 <1>       ;     NOTE: If the process is running while the time count
2336                                 <1>       ;           is reached, kernel will put signal return (response)
2337                                 <1>       ;           byte value at requested address during timer
2338                                 <1>       ;           interrupt and the process will continue to run.
2339                                 <1>       ;           Program/process must call (jump to) it's timer event
2340                                 <1>       ;           function as required, for checking the timer event
2341                                 <1>       ;           status via signal return (response) byte address.
2342                                 <1>       ;
2343                                 <1>       ;           If the process is not running (waiting or sleeping
2344                                 <1>       ;           or released) while the time count is reached,
2345                                 <1>       ;           it is restarted from where it left, to ensure
2346                                 <1>       ;           proper multi media (video, audio, clock, timer)
2347                                 <1>       ;           functionality.
2348                                 <1>       ;
2349                                 <1>       ;           (It is better to use 'syswait' or 'syssleep',
2350                                 <1>       ;           or 'sysrele' system call just after the timer
2351                                 <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 0000E14A C605[60930100]00      <1>       mov   byte [tcallback], 0
2368 0000E151 C605[61930100]00      <1>       mov   byte [trtc], 0
2369 0000E158 C705[D0030300]0000-   <1>       mov   dword [u.tcb], 0 ; this is not necessary...
2369 0000E160 0000                  <1>
2370                                 <1>
2371 0000E162 80FF80                <1>       cmp   bh, 80h
2372 0000E165 7225                  <1>       jb    short systimer_cb2
2373 0000E167 7704                  <1>       ja    short systimer_cb0
2374                                 <1>
2375 0000E169 31D2                  <1>       xor   edx, edx ; 0, reset callback address
2376 0000E16B EB0B                  <1>       jmp   short systimer_cb1
2377                                 <1>
2378                                 <1> systimer_cb0:
2379 0000E16D 80FF84                <1>       cmp   bh, 84h
```

```
2380 0000E170 7764               <1>       ja     short systimer_5 ;  undefined, error
2381                             <1>
2382                             <1>       ;mov   byte [tcallback], 1 ; 19/12/2016
2383 0000E172 FE05[60930100]     <1>       inc    byte [tcallback]
2384                             <1>
2385                             <1> systimer_cb1:
2386 0000E178 0FB635[B3030300]   <1>       movzx  esi, byte [u.uno] ; process number
2387 0000E17F 66C1E602           <1>       shl    si, 2
2388 0000E183 8996[0C010300]     <1>       mov    [esi+p.tcb-4], edx ; set process timer callback address
2389                             <1>                                 ; (overwrite prev value if it is set!)
2390 0000E189 80E77F             <1>       and    bh, 7Fh
2391                             <1>
2392                             <1> systimer_cb2:
2393 0000E18C 80FF02             <1>       cmp    bh, 2
2394 0000E18F 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 0000E191 774B               <1>         ja      short systimer_6
2398                             <1>
2399 0000E193 20FF               <1>       and    bh, bh
2400 0000E195 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 0000E19B B00A               <1>       mov    al, 10 ; (*)
2406                             <1>
2407                             <1> systimer_0:
2408 0000E19D B710               <1>       mov    bh, 16
2409                             <1>       ;
2410 0000E19F 383D[238E0100]     <1>       cmp    [timer_events], bh ; 16 ; 07/06/2016
2411 0000E1A5 7319               <1>       jnb    short systimer_3  ; max. 16 timer events
2412                             <1>       ;
2413 0000E1A7 50                 <1>       push   eax ; (*)
2414                             <1>
2415 0000E1A8 BF[60040300]       <1>       mov    edi, timer_set  ; beginning address of timer events
2416                             <1>                              ; setting space
2417 0000E1AD 30C0               <1>       xor    al, al ; 0
2418                             <1> systimer_1:
2419 0000E1AF FEC0               <1>       inc    al
2420 0000E1B1 803F00             <1>       cmp    byte [edi], 0       ; is it free space ?
2421 0000E1B4 7639               <1>       jna    short systimer_7 ; yes
2422 0000E1B6 FECF               <1>       dec    bh
2423 0000E1B8 7405               <1>       jz     short systimer_2
2424 0000E1BA 83C710             <1>       add    edi, 16
2425 0000E1BD EBF0               <1>       jmp    short  systimer_1 ; next event space
2426                             <1>
2427                             <1> systimer_2:
2428 0000E1BF 58                 <1>       pop    eax ; (*) discard
2429                             <1> systimer_3:
2430 0000E1C0 C605[64030300]00   <1>       mov    byte [u.r0], 0
2431                             <1> systimer_4:
2432 0000E1C7 C705[C8030300]1B00- <1>         mov     dword [u.error], ERR_MISC
2432 0000E1CF 0000               <1>
2433                             <1>                                   ; one of miscellaneous/other errors
2434 0000E1D1 E9F6F6FFFF         <1>       jmp    error ; cf -> 1
2435                             <1>
2436                             <1> systimer_5:
2437 0000E1D6 883D[64030300]     <1>       mov    [u.r0], bh ; Time count unit (=2 or >3)
2438 0000E1DC EBE9               <1>       jmp    short systimer_4 ; 07/06/2016
2439                             <1>
2440                             <1> systimer_6:
2441 0000E1DE 80FF04             <1>       cmp    bh, 4
2442 0000E1E1 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 0000E1E3 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 0000E1E5 B0B6               <1>       mov    al, 182 ; (*) ; 18.2 * 10
2458 0000E1E7 FE05[61930100]     <1>       inc    byte [trtc] ; timer event via real time clock
2459 0000E1ED EBAE               <1>         jmp     short systimer_0
2460                             <1>
2461                             <1> systimer_7:
2462 0000E1EF A2[64030300]       <1>       mov    [u.r0], al ; timer event number
2463                             <1>       ;
2464                             <1>       ; edi = address of empty timer event area
2465 0000E1F4 A0[B3030300]       <1>       mov    al, [u.uno]
2466 0000E1F9 FA                 <1>       cli    ; disable interrupts
2467 0000E1FA AA                 <1>       stosb  ; process number
2468 0000E1FB A0[60930100]       <1>       mov    al, [tcallback] ; timer callback flag
2469 0000E200 AA                 <1>       stosb  ; 1= callback method, 0= signal response byte method
2470 0000E201 A0[61930100]       <1>       mov    al, [trtc] ; timer interrupt type
2471 0000E206 AA                 <1>       stosb  ; 1= real time clock, 0= programmable interval timer
2472 0000E207 88D8               <1>       mov    al, bl ; Signal return (Response) value
2473 0000E209 AA                 <1>       stosb   ; response byte
2474 0000E20A 58                 <1>       pop    eax ; (*) ; 10 or 182
2475 0000E20B 89D3               <1>       mov    ebx, edx ; virtual address for response/signal byte
2476 0000E20D F7E1               <1>       mul    ecx
2477                             <1>       ; (eax = 10 * count of 18.2 Hz timer ticks)
2478                             <1>       ; (count down step = 10)
2479 0000E20F AB                 <1>       stosd  ; count limit (reset value)
2480 0000E210 AB                 <1>       stosd  ; current count value
2481                             <1>
2482                             <1>       ; 19/12/2016
2483 0000E211 803D[60930100]00   <1>       cmp    byte [tcallback], 0 ; timer callback method ?
```

```
2484 0000E218 7604              <1>       jna    short systimer_17 ; no
2485 0000E21A 89D8              <1>       mov    eax, ebx ; virtual address for callback routine
2486 0000E21C 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 0000E21E FE05[62930100]    <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 0000E224 E8B580FFFF        <1>       call   get_physical_addr
2497 0000E229 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 0000E22B AB                <1>       stosd  ; response addr (physical) or callback addr (virtual)
2501 0000E22C FE05[238E0100]    <1>       inc    byte [timer_events] ; 07/06/201
2502                            <1>       ; 02/01/2017
2503 0000E232 0FB605[B3030300]  <1>       movzx  eax, byte [u.uno]
2504 0000E239 FE80[FF000300]    <1>       inc    byte [eax+p.timer-1]
2505                            <1>       ;
2506 0000E23F FB                <1>       sti    ; enable interrupts
2507 0000E240 E9A7F6FFFF        <1>       jmp    sysret
2508                            <1>
2509                            <1> systimer_8:
2510                            <1>       ; 10/06/2016
2511                            <1>       ; 07/06/2016
2512 0000E245 28C0              <1>       sub    al, al ; 0
2513 0000E247 8847F4            <1>       mov    [edi-12], al ; clear process number (free timer event)
2514                            <1>       ;mov   dword [edi], eax ; 0
2515 0000E24A FB                <1>       sti
2516 0000E24B A2[64030300]      <1>       mov    [u.r0], al ; 0
2517 0000E250 E977F6FFFF        <1>       jmp    error
2518                            <1>
2519                            <1> systimer_9:
2520                            <1>       ; 10/06/2016
2521                            <1>       ; 07/06/2016
2522 0000E255 28C0              <1>       sub    al, al
2523 0000E257 A2[64030300]      <1>       mov    byte [u.r0], al ; 0
2524 0000E25C 3805[238E0100]    <1>       cmp     byte [timer_events], al ;  0
2525 0000E262 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 0000E264 BE[60040300]      <1>       mov    esi, timer_set  ; beginning address of timer events
2531                            <1>                             ; setting space
2532 0000E269 A0[B3030300]      <1>       mov    al, [u.uno]
2533                            <1>
2534 0000E26E B710              <1>       mov    bh, 16
2535                            <1>
2536 0000E270 08DB              <1>       or     bl, bl
2537 0000E272 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 0000E274 FA                <1>       cli    ; disable interrupts
2542                            <1> systimer_10:
2543                            <1>       ; 10/06/2016
2544                            <1>       ; 07/06/2016
2545 0000E275 8A26              <1>       mov    ah, [esi]
2546 0000E277 08E4              <1>       or     ah, ah ; 0 ?
2547 0000E279 7411              <1>       jz     short systimer_11
2548 0000E27B 38C4              <1>       cmp    ah, al ; is the process number (owner) same ?
2549 0000E27D 750D              <1>         jne     short systimer_11 ; no
2550                            <1>
2551                            <1>       ;mov   byte [esi], 0
2552 0000E27F 66C7060000        <1>       mov    word [esi], 0 ; clear
2553                            <1>       ;mov   dword [esi+12], 0 ; clear
2554                            <1>
2555 0000E284 FE0D[238E0100]    <1>       dec    byte [timer_events]
2556 0000E28A 7409              <1>       jz     short systimer_12
2557                            <1>
2558                            <1> systimer_11:
2559 0000E28C FECF              <1>       dec    bh
2560 0000E28E 7405              <1>       jz     short systimer_12
2561 0000E290 83C610            <1>       add    esi, 16
2562 0000E293 EBE0              <1>       jmp    short systimer_10
2563                            <1>
2564                            <1> systimer_12:
2565 0000E295 0FB635[B3030300]  <1>       movzx  esi, byte [u.uno]
2566 0000E29C 08DB              <1>       or     bl, bl ; all timer events or one timer event ?
2567 0000E29E 740C              <1>       jz     short systimer_13
2568 0000E2A0 8A9E[FF000300]    <1>       mov    bl, [esi+p.timer-1]
2569 0000E2A6 20DB              <1>       and    bl, bl ; previous number of timer events for the process
2570 0000E2A8 7408              <1>       jz     short systimer_14
2571 0000E2AA FECB              <1>       dec    bl  ; previous number of timer events for the process - 1
2572                            <1> systimer_13:
2573 0000E2AC 889E[FF000300]    <1>       mov    [esi+p.timer-1], bl ; 0 ; no timer events for process
2574                            <1> systimer_14:
2575 0000E2B2 FB                <1>       sti    ; enable interrupts
2576 0000E2B3 E934F6FFFF        <1>       jmp    sysret
2577                            <1>
2578                            <1> systimer_15:
2579 0000E2B8 38FB              <1>       cmp    bl, bh ; 16
2580 0000E2BA 0F8707FFFFFF      <1>         ja      systimer_4      ; max. 16 timer events !
2581                            <1>       ;
2582 0000E2C0 88DA              <1>       mov    dl, bl
2583 0000E2C2 FECA              <1>       dec    dl  ; 16 -> 15 ... 1 -> 0
2584 0000E2C4 C0E204            <1>       shl    dl, 4 ; * 16
2585 0000E2C7 0FB6FA            <1>       movzx  edi, dl
2586 0000E2CA 01F7              <1>       add    edi, esi ; timer_set
2587                            <1>
2588 0000E2CC 3A07              <1>       cmp    al, [edi] ; process number
```

```
0000E2CE 0F85F3FEFFFF        <1>         jne     systimer_4
                             <1>
                             <1>       ; same process ID
0000E2D4 FA                  <1>       cli    ; disable interrupts
                             <1>       ; 10/06/2016 ; 02/01/2017
                             <1>       ;mov   byte [edi], 0
0000E2D5 66C7070000          <1>       mov    word [edi], 0 ; clear
                             <1>       ;mov   dword [edi+12], 0 ; clear
0000E2DA FE0D[238E0100]      <1>       dec    byte [timer_events]
0000E2E0 EBB3                <1>       jmp    short systimer_12
                             <1>
                             <1> sysvideo: ; VIDEO DATA TRANSFER FUNCTIONS
                             <1>       ; 24/01/2021
                             <1>       ; 23/01/2021
                             <1>       ; 22/01/2021
                             <1>       ; 18/01/2021, 19/01/2021
                             <1>       ; 04/01/2021, 10/01/2021, 11/01/2021
                             <1>       ; 01/01/2021, 02/01/2021, 03/01/2021
                             <1>       ; 28/12/2020, 29/12/2020, 30/12/2020
                             <1>       ; 25/12/2020, 26/12/2020
                             <1>       ; 21/12/2020, 23/12/2020
                             <1>       ; 12/12/2020, 14/12/2020
                             <1>       ; 10/12/2020, 11/12/2020
                             <1>       ; 03/12/2020, 04/12/2020
                             <1>       ; 22/11/2020, 23/11/2020
                             <1>       ; 21/11/2020 (TRDOS 386 v2.0.3)
                             <1>       ; 12/05/2017
                             <1>       ; 11/07/2016
                             <1>       ; 13/06/2016
                             <1>       ; 16/05/2016 - TRDOS 386 (TRDOS v2.0)
                             <1>       ;
                             <1>       ; VIDEO DATA TRANSFER FUNCTIONS:
                             <1>       ;
                             <1>       ; Inputs:
                             <1>       ;     BH = 0 = VIDEO BIOS Mode 3, tty/text mode data transfers
                             <1>       ;          BL =
                             <1>       ;            Bits 0&1, Transfer direction
                             <1>       ;                  0 - System to system
                             <1>       ;                  1 - User to system
                             <1>       ;                  2 - System to user
                             <1>       ;                  3 - User to user (undefined)
                             <1>       ;            Bits 2&3, Transfer Type
                             <1>       ;                  0 - Display page transfer
                             <1>       ;                  1 - Display page window transfer
                             <1>       ;                  2 - Frame/Viewport/Window address transfer
                             <1>       ;                  3 - Window handle transfer (undefined)
                             <1>       ;            ; 23/11/2020
                             <1>       ;            Bits 4..7 - Reserved, undefined (must be 0)
                             <1>       ;
                             <1>       ;          /// BL = 0 -> System to system (display page) transfer
                             <1>       ;             CL = Source page
                             <1>       ;             DL = Destination page
                             <1>       ;          /// BL = 1&2 -> user to system & system to user transfer
                             <1>       ;             ECX = User' buffer address
                             <1>       ;             DL = Video page
                             <1>       ;          /// BL = 5&6 -> user to system, system to user transfer
                             <1>       ;            (system window is in current/active display page)
                             <1>       ;             ESI = User's buffer address
                             <1>       ;             ECX Low 16 bits = Top left column (X1 position)
                             <1>       ;             ECX High 16 bits = Top row (Y1 position)
                             <1>       ;             EDX Low 16 bits = Bottom right column (X2 position)
                             <1>       ;             EDX High 16 bits = Bottom row (Y2 position)
                             <1>       ;               If BL = 5 ->
                             <1>       ;             EDI = Swap address (in user's memory space)
                             <1>       ;             (If swap address > 0, previous content of the window
                             <1>       ;             will be saved into swap area in user's memory space)
                             <1>       ;          /// BL = 4 -> system to system transfer
                             <1>       ;             ESI = System's source buffer (video page) address
                             <1>       ;             ECX Low 16 bits = Top left column (X1 position)
                             <1>       ;             ECX High 16 bits = Top row (Y1 position)
                             <1>       ;             EDX Low 16 bits = Bottom right column (X2 position)
                             <1>       ;             EDX High 16 bits = Bottom row (Y2 position)
                             <1>       ;             EDI = System's destination buffer (video page) address
                             <1>       ;            ; 22/11/2020
                             <1>       ;          /// BL = 9&10 -> user to system, system to user transfer
                             <1>       ;            (system window is in current/active display page)
                             <1>       ;             ESI = User's buffer address
                             <1>       ;              CX = offset in video page (char position in 80*25)
                             <1>       ;              DX = transfer count (character count, max. 80*25)
                             <1>       ;               If BL = 5 ->
                             <1>       ;             EDI = Swap address (in user's memory space)
                             <1>       ;             (If swap address > 0, previous content of the window
                             <1>       ;             will be saved into swap area in user's memory space)
                             <1>       ;          /// BL = 8 -> system to system transfer
                             <1>       ;             ESI = System's source buffer (video page) address
                             <1>       ;              CX = offset in video page (char position in 80*25)
                             <1>       ;              DX = transfer count (character count, max. 80*25)
                             <1>       ;             EDX High 16 bits = Bottom row (Y2 position)
                             <1>       ;             EDI = System's destination buffer (video page) address
                             <1>       ;
                             <1>       ;            ; 23/11/2020 (major modification)
                             <1>       ;            ; 22/11/2020 (bugfixes and extensions)
                             <1>       ;     BH = 1 = VGA Graphics (0A0000h) data transfers
                             <1>       ;          BL =
                             <1>       ;            x0h = Fill color (color in CL) (64K)
                             <1>       ;            x1h = User to system display page transfer
                             <1>       ;            x2h = System to user display page transfer
                             <1>       ;            x3h = NOT bits in window (ECX, EDX)
                             <1>       ;            x4h = Window copy (system to system)
                             <1>       ;            x5h = User to system window transfer
                             <1>       ;            x6h = System to user window transfer
                             <1>       ;            x7h = AND display page bytes with CL
                             <1>       ;            x8h = OR display page bytes with CL
                             <1>       ;            x9h = XOR display page bytes with CL
                             <1>       ;            ; 22/11/2020 (TRDOS v2.0.3)
```

```
2694                                    <1>       ;             xAh = INC display page bytes
2695                                    <1>       ;             xBh = DEC display page bytes
2696                                    <1>       ;             xCh = NEG display page bytes
2697                                    <1>       ;             xDh = NOT display page bytes
2698                                    <1>       ;            ; 01/01/2021
2699                                    <1>       ;            xEh = masked window write
2700                                    <1>       ;            ; 02/01/2021
2701                                    <1>       ;            xFh = write character (font)
2702                                    <1>       ;
2703                                    <1>       ;            x = 0 -> screen width = 320
2704                                    <1>       ;            x = 1 -> screen width = 640
2705                                    <1>       ;            x = 2 -> screen width = 800
2706                                    <1>       ;
2707                                    <1>       ;          /// BL = 0 -> Fill color (all screen pixels)
2708                                    <1>       ;             CL = Color value,
2709                                    <1>       ;             If CH > 0, color in CL will be mixed with pixel color
2710                                    <1>       ;          /// BL = 1&2 -> user to system & system to user transfer
2711                                    <1>       ;             ECX = User buffer
2712                                    <1>       ;          /// BL = 5&6 -> user to system, system to user transfer
2713                                    <1>       ;            (window in current display page and in current mode)
2714                                    <1>       ;             ESI = User's buffer address
2715                                    <1>       ;             ECX Low 16 bits = Top left column (X1 position)
2716                                    <1>       ;             ECX High 16 bits = Top row (Y1 position)
2717                                    <1>       ;             EDX Low 16 bits = Bottom right column (X2 position)
2718                                    <1>       ;             EDX High 16 bits = Bottom row (Y2 position)
2719                                    <1>                 ;           /// BL = 4 -> system to system (window) transfer
2720                                    <1>       ;             ESI = System's source buffer (video page) address
2721                                    <1>       ;             ECX Low 16 bits = Top left column (X1 position)
2722                                    <1>       ;             ECX High 16 bits = Top row (Y1 position)
2723                                    <1>       ;             EDX Low 16 bits = Bottom right column (X2 position)
2724                                    <1>       ;             EDX High 16 bits = Bottom row (Y2 position)
2725                                    <1>       ;             EDI = System's destination buffer (video page) address
2726                                    <1>       ;          /// BL = 3 -> NOT byte in display page/memory
2727                                    <1>       ;             ECX Low 16 bits = Top left column (X1 position)
2728                                    <1>       ;             ECX High 16 bits = Top row (Y1 position)
2729                                    <1>       ;             EDX Low 16 bits = Bottom right column (X2 position)
2730                                    <1>       ;             EDX High 16 bits = Bottom row (Y2 position)
2731                                    <1>       ;          /// BL = 7,8,9 -> Display page pixel operations
2732                                    <1>       ;                          (and, or, xor)
2733                                    <1>       ;             CL = color
2734                                    <1>       ;            ; 22/11/2020
2735                                    <1>       ;          /// BL = 10..13 -> Display page pixel operations
2736                                    <1>       ;                          (inc, dec, neg, not)
2737                                    <1>       ;          /// BL = 14 -> Masked color change
2738                                    <1>       ;             ESI = User's mask buffer address
2739                                    <1>       ;             ECX Low 16 bits = Top left column (X1 position)
2740                                    <1>       ;             ECX High 16 bits = Top row (Y1 position)
2741                                    <1>       ;             EDX Low 16 bits = Bottom right column (X2 position)
2742                                    <1>       ;             EDX High 16 bits = Bottom row (Y2 position)
2743                                    <1>       ;             EDI = (Mask) Color
2744                                    <1>       ;                EBX bit 16 to 18 -> mask operation/option
2745                                    <1>       ;                   0 = new color (in edi)
2746                                    <1>       ;                   1 = add color (in edi) -up to 0FFh-
2747                                    <1>       ;                   2 = average color (in edi) -(current+edi)/2-
2748                                    <1>       ;                   3 = sub color (in edi) -down to 0-
2749                                    <1>       ;                   4 = inc      color -up to 0FFh-
2750                                    <1>       ;                   5 = dec      color -down to 0-
2751                                    <1>       ;                   6 = not color
2752                                    <1>       ;                   7 = neg color
2753                                    <1>       ;          /// BL = 15 -> Write character (font) -in current mode-
2754                                    <1>       ;             ECX Low 16 bits = Top left column (X1 position)
2755                                    <1>       ;             ECX High 16 bits = Top row (Y1 position)
2756                                    <1>       ;              DL = Character's ASCII code
2757                                    <1>       ;              DH = Character's color
2758                                    <1>       ;                EBX bit 16 -> character size option
2759                                    <1>       ;                   0 = 8x16 character font
2760                                    <1>       ;                   1 = 8x8 character font
2761                                    <1>       ;                EBX bit 17 & 18 -> scale
2762                                    <1>       ;                   0 = 1/1 (8 pixels per char row)
2763                                    <1>       ;                   1 = 2/1 (16 pixels per char row)
2764                                    <1>       ;                   2 = 4/1 (32 pixels per char row)
2765                                    <1>       ;                   3 = 8/1 (64 pixels per char row)
2766                                    <1>       ;             ; 05/01/2021
2767                                    <1>       ;                 EBX bit 19 -> user font option (1 = user font)
2768                                    <1>       ;
2769                                    <1>       ;      BH = 2 = Super VGA, LINEAR FRAME BUFFER data transfers
2770                                    <1>       ;           BL =
2771                                    <1>       ;             0 = Fill color (color in ECX) (vbe mode screen size)
2772                                    <1>       ;             1 = User to system display page transfer
2773                                    <1>       ;             2 = System to user display page transfer
2774                                    <1>       ;             3 = NOT bits in window (ECX, EDX)
2775                                    <1>       ;             4 = Window copy (system to system)
2776                                    <1>       ;             5 = User to system window transfer
2777                                    <1>       ;             6 = System to user window transfer
2778                                    <1>       ;             7 = AND display page bytes with ECX
2779                                    <1>       ;             8 = OR display page bytes with ECX
2780                                    <1>       ;             9 = XOR display page bytes with ECX
2781                                    <1>       ;             ; 22/11/2020 (TRDOS v2.0.3)
2782                                    <1>       ;              10 = INC display page bytes
2783                                    <1>       ;              11 = DEC display page bytes
2784                                    <1>       ;              12 = NEG display page bytes
2785                                    <1>       ;              13 = NOT display page bytes
2786                                    <1>       ;             ; 01/01/2021
2787                                    <1>       ;             14 = masked window write
2788                                    <1>       ;             ; 02/01/2021
2789                                    <1>       ;             15 = write character (font)
2790                                    <1>       ;
2791                                    <1>       ;             ; 25/12/2020
2792                                    <1>       ;             High word of EBX contain VESA VBE mode for resolution
2793                                    <1>       ;             (if hw of ebx = 0, current vesa vbe mode will be used)
2794                                    <1>       ;
2795                                    <1>       ;          /// BL = 0 -> Fill color (all screen pixels)
2796                                    <1>       ;             CL = Color value,
2797                                    <1>       ;             If CH > 0, color in CL will be mixed with pixel color
2798                                    <1>       ;          /// BL = 1&2 -> user to system & system to user transfer
```

```
2799                    <1>     ;           ECX = User buffer
2800                    <1>     ;         /// BL = 5&6 -> user to system, system to user transfer
2801                    <1>     ;           (window in current display page and in current mode)
2802                    <1>     ;           ESI = User's buffer address
2803                    <1>     ;           ECX Low 16 bits = Top left column (X1 position)
2804                    <1>     ;           ECX High 16 bits = Top row (Y1 position)
2805                    <1>     ;           EDX Low 16 bits = Bottom right column (X2 position)
2806                    <1>     ;           EDX High 16 bits = Bottom row (Y2 position)
2807                    <1>     ;         /// BL = 4 -> system to system (window) transfer
2808                    <1>     ;           ESI = System's source buffer (video page) address
2809                    <1>     ;           ECX Low 16 bits = Top left column (X1 position)
2810                    <1>     ;           ECX High 16 bits = Top row (Y1 position)
2811                    <1>     ;           EDX Low 16 bits = Bottom right column (X2 position)
2812                    <1>     ;           EDX High 16 bits = Bottom row (Y2 position)
2813                    <1>     ;           EDI = System's destination buffer (video page) address
2814                    <1>     ;         /// BL = 3 -> NOT byte in display page/memory
2815                    <1>     ;           ECX Low 16 bits = Top left column (X1 position)
2816                    <1>     ;           ECX High 16 bits = Top row (Y1 position)
2817                    <1>     ;           EDX Low 16 bits = Bottom right column (X2 position)
2818                    <1>     ;           EDX High 16 bits = Bottom row (Y2 position)
2819                    <1>     ;         /// BL = 7,8,9 -> Display page pixel operations
2820                    <1>     ;                         (and, or, xor)
2821                    <1>     ;           CL = color (for 8bpp)
2822                    <1>     ;           CX = color (for 16bpp)
2823                    <1>     ;          ECX = color (for 24bpp, 24 bits of ECX are used)
2824                    <1>     ;          ECX = color (for 32bpp)
2825                    <1>     ;          ; 22/11/2020
2826                    <1>     ;         /// BL = 10..13 -> Display page pixel operations
2827                    <1>     ;                           (inc, dec, neg, not)
2828                    <1>     ;         /// BL = 14 -> Masked color change
2829                    <1>     ;           ESI = User's mask buffer address
2830                    <1>     ;           ECX Low 16 bits = Top left column (X1 position)
2831                    <1>     ;           ECX High 16 bits = Top row (Y1 position)
2832                    <1>     ;           EDX Low 16 bits = Bottom right column (X2 position)
2833                    <1>     ;           EDX High 16 bits = Bottom row (Y2 position)
2834                    <1>     ;           EDI = (Mask) Color
2835                    <1>     ;              EBX bit 16 to 18 -> mask operation/option
2836                    <1>     ;                 0 = new color (in edi)
2837                    <1>     ;                 1 = add color (in edi) -up to 0FFFFFFFFh-
2838                    <1>     ;                 2 = average color (in edi) -(current+edi)/2-
2839                    <1>     ;                 3 = sub color (in edi) -down to 0-
2840                    <1>     ;                 4 = inc      color -up to 0FFFFFFFFh-
2841                    <1>     ;                 5 = dec      color -down to 0-
2842                    <1>     ;                 6 = not color
2843                    <1>     ;                 7 = neg color
2844                    <1>     ;         /// BL = 15 -> Write character (font) -in current mode-
2845                    <1>     ;           ECX Low 16 bits = Top left column (X1 position)
2846                    <1>     ;           ECX High 16 bits = Top row (Y1 position)
2847                    <1>     ;            DL = Character's ASCII code
2848                    <1>     ;            DH = Character's color
2849                    <1>     ;              EBX bit 16 -> character size option
2850                    <1>     ;                 0 = 8x16 character font
2851                    <1>     ;                 1 = 8x8 character font
2852                    <1>     ;              EBX bit 17 & 18 -> scale
2853                    <1>     ;                 0 = 1/1 (8 pixels per char row)
2854                    <1>     ;                 1 = 2/1 (16 pixels per char row)
2855                    <1>     ;                 2 = 4/1 (32 pixels per char row)
2856                    <1>     ;                 3 = 8/1 (64 pixels per char row)
2857                    <1>     ;           ; 05/01/2021
2858                    <1>     ;              EBX bit 19 -> user font option (1 = user font)
2859                    <1>     ;
2860                    <1>     ;           Example: (21/11/2020)
2861                    <1>     ;                 esi = 0B0000h (window start value)
2862                    <1>     ;                 ecx = 00000000h (start at row 0, column 0)
2863                    <1>     ;                 edx = 0018004Fh (end at row 24,column 79)
2864                    <1>     ;                 row count= 25, column count = 80
2865                    <1>     ;                 (end) byte offset = (80*2)*24+(79*2) = 3998
2866                    <1>     ;                 end value of esi = 0B8F9Eh (last word of page)
2867                    <1>     ;                 (! end address is included to transfer !)
2868                    <1>     ;                 ((just as line start and line end coordinates))
2869                    <1>     ;
2870                    <1>     ; Outputs:
2871                    <1>     ;     EAX = transfer/byte count
2872                    <1>     ;
2873                    <1>     ;     NOTE: If the source or destination address passes out of
2874                    <1>     ;     video pages (display memory limits), data will not be transferred
2875                    <1>     ;     and EAX will return as 0.
2876                    <1>     ;
2877                    <1>     ; 04/01/2021
2878                    <1>     ; PIXEL READ/WRITE (in current/active video mode)
2879                    <1>     ;
2880                    <1>     ;     BH = 3 = Read/Write pixel(s) -for all graphics modes-
2881                    <1>     ;          BL =
2882                    <1>     ;            0 = Read pixel
2883                    <1>     ;            1 = Write pixel
2884                    <1>     ;            2 = swap pixel colors
2885                    <1>     ;            3 = mix pixel colors
2886                    <1>     ;
2887                    <1>     ;           > 3 = invalid/unimplemented
2888                    <1>     ;
2889                    <1>     ;          CL = color for writing pixel(s) or
2890                    <1>     ;          ECX = color for writing pixel(s) in true color modes
2891                    <1>     ;
2892                    <1>     ;          EDX = Offset from start of video memory (0A0000h)
2893                    <1>     ;               or start of linear frame buffer
2894                    <1>     ; Note:
2895                    <1>     ;     Pixel read/write will be performed in current video mode.
2896                    <1>     ;     If [CRT_MODE] < 0FFh, 0A0000h will be used
2897                    <1>     ;        as video memory and limit will be 65536
2898                    <1>     ;        (new/mix pixel color will be in CL)
2899                    <1>     ;     if [CRT_MODE] = 0FFh (VESA VBE video mode)
2900                    <1>     ;        LFB base address will be used as video memory
2901                    <1>     ;        and limit will be video page size
2902                    <1>     ;        (new/mix pixel color will be in CL)
2903                    <1>     ;
```

```
2904                                <1>       ; Outputs:
2905                                <1>       ;     EAX = pixel color (according to BL and ECX input)
2906                                <1>       ;     EAX = 0 (pixel color is 0 or there is an error)
2907                                <1>       ;          (BL will return as 0FFh if there is an error)
2908                                <1>       ;
2909                                <1>       ; DIRECT (STANDARD VGA/CGA) DISPLAY MEMORY ACCESS FUNCTIONS:
2910                                <1>       ;
2911                                <1>       ;     BH = 4 = CGA direct video memory (0B8000h, 32K) access
2912                                <1>       ;            Page directory & page tables of the user's
2913                                <1>       ;            program will be updated to direct access to
2914                                <1>       ;            0B8000h (32K) video (CGA, color) memory; if
2915                                <1>       ;            there is not a permission conflict or lock!
2916                                <1>       ;              (User's program/process will have permission to
2917                                <1>       ;            access locked display memory if the owner is
2918                                <1>       ;            it's parent.)
2919                                <1>         ;
2920                                <1>       ;         Screen width = 320
2921                                <1>       ;
2922                                <1>       ;     BH = 5 = VGA direct video memory (0A0000h, 64K) access
2923                                <1>       ;            Page directory & page tables of the user's
2924                                <1>       ;            program will be updated to direct access to
2925                                <1>       ;            0A0000h (64K) video (VGA) memory; if there is not
2926                                <1>       ;            a permission conflict or lock!
2927                                <1>       ;              (User's program/process will have permission to
2928                                <1>       ;            access locked display memory if the owner is
2929                                <1>       ;            it's parent.)
2930                                <1>       ;
2931                                <1>       ;         ; 23/11/2020
2932                                <1>       ;         Screen width options = 320, 640, 800
2933                                <1>       ;
2934                                <1>       ; Outputs:
2935                                <1>       ;     EAX = Display memory address for direct access
2936                                <1>       ;           0A0000h for VGA, 0B8000h for CGA
2937                                <1>       ;     (Display memory size: 32K for CGA, 64K for VGA)
2938                                <1>       ;     EAX = 0 if display page access permission has been denied.
2939                                <1>       ;           (Locked!)
2940                                <1>       ;
2941                                <1>       ; LINEAR FRAME BUFFER ACCESS FUNCTIONS:
2942                                <1>       ;
2943                                <1>       ;     BH = 6 = Linear Frame Buffer direct video memory access
2944                                <1>       ;
2945                                <1>       ;            Page directory & page tables of the user's
2946                                <1>       ;            program will be updated to direct access to
2947                                <1>       ;            the configured LFB (Linear Frame Buffer) address,
2948                                <1>       ;            if there is not a permission conflict or lock!
2949                                <1>       ;              (User's program/process will have permission to
2950                                <1>       ;            access locked display memory if the owner is
2951                                <1>       ;            it's parent.)
2952                                <1>       ;
2953                                <1>       ;        ; 10/12/2020
2954                                <1>       ;        BL = 0FFh -> Direct LFB access for current video mode
2955                                <1>       ;        BL = XXh < 0FFh -> Direct LFB access
2956                                <1>       ;                           for VESA video mode 1XXh
2957                                <1>       ;
2958                                <1>       ;            Return: EAX = Linear Frame Buffer address
2959                                <1>       ;                   (EAX = 0 -> error)
2960                                <1>       ;                 If EAX > 0
2961                                <1>       ;                   EDX = Frame Buffer Size in bytes
2962                                <1>       ;                      BH = Requested Video Mode - 100h
2963                                <1>       ;                           (VESA VBE video modes)
2964                                <1>       ;                      BL = bits per pixel
2965                                <1>       ;                          8 = 256 colors, 8
2966                                <1>       ;                           16 = 65536 colors, 5-6(G)-5
2967                                <1>       ;                           24 = RGB, 16M colors, 8-8-8
2968                                <1>       ;                           32 = RGB + alpha bytes, 8-8-8-8
2969                                <1>       ;                 If BH = 0FFh
2970                                <1>       ;                    BL = VGA/CGA video mode (also EAX = 0)
2971                                <1>       ;
2972                                <1>       ;            ** Function will return with EAX = 0 if the mode
2973                                <1>       ;            is not a valid VESA VBE video mode as 1??h **
2974                                <1>       ;
2975                                <1>       ;            ECX = Pixel resolution
2976                                <1>       ;                  CX = Width (320, 640, 800, 1024, 1366, 1920)
2977                                <1>       ;                  High 16 bits of ECX = Height
2978                                <1>       ;
2979                                <1>       ;     23/11/2020
2980                                <1>       ; ***  GET VIDEO MODE & LINEAR FRAME BUFFER INFO
2981                                <1>       ;     (This function -7- also is used for VGA and CGA modes)
2982                                <1>       ;
2983                                <1>       ;     BH = 7 = Get Linear Frame Buffer info
2984                                <1>       ;
2985                                <1>       ;        ; 22/01/2021
2986                                <1>       ;        ; 10/12/2020
2987                                <1>       ;        BL = any -not used- (22/01/2021)
2988                                <1>       ;
2989                                <1>       ;           Return:
2990                                <1>       ;           EAX = Frame Buffer Address (0 = is not in use)
2991                                <1>       ;           EDX = Frame Buffer Size in bytes
2992                                <1>       ;            BH = Current Video Mode - 100h  ; 22/01/2021
2993                                <1>       ;                (VESA VBE video modes)
2994                                <1>       ;            BL = bits per pixel
2995                                <1>       ;                 8 = 256 colors, 8
2996                                <1>       ;                  16 = 65536 colors, 5-6(G)-5
2997                                <1>       ;                  24 = RGB, 16M colors, 8-8-8
2998                                <1>       ;                  32 = RGB + alpha bytes, 8-8-8-8
2999                                <1>       ;           If BH = 0FFh
3000                                <1>       ;              BL = VGA/CGA video mode (also EAX = 0)
3001                                <1>       ;
3002                                <1>       ;           Note:
3003                                <1>       ;           Alpha byte will be used as virtual color index.
3004                                <1>       ;           (32 bit pixel colors.. byte 0,1,2 rgb and 3 alpha)
3005                                <1>       ;
3006                                <1>       ;           ** Function will return with EAX = 0 if the mode
3007                                <1>       ;           is not a valid VESA VBE video mode as 1??h **
3008                                <1>       ;
```

```
3009                                    <1>     ;           ECX = Pixel resolution
3010                                    <1>     ;                 CX = Width (320, 640, 800, 1024, 1366, 1920)
3011                                    <1>     ;                 High 16 bits of ECX = Height
3012                                    <1>     ;
3013                                    <1>     ;     NOTE: Each process will have it's own frame buffer
3014                                    <1>     ;           address and resolution parameters in 'u' area.
3015                                    <1>     ;           Then, if the current frame buffer & resolution
3016                                    <1>     ;           is different, frame buffer r/w functions
3017                                    <1>     ;           will use scale factor to convert process's
3018                                    <1>       ;           pixel coordinates to actual screen coordinates.
3019                                    <1>     ;           resolution -> dimensional scale
3020                                    <1>     ;           color size -> color scale
3021                                    <1>     ;          * RGB (TRUE) colors to 256 colors conversion:
3022                                    <1>       ;           TRUE Colors -> 8,8,8 (R,G,B; byte 0 is R)
3023                                    <1>     ;           256 colors -> 2,2,2,2 (R,G,B,L; bit 0&1 is R)
3024                                    <1>     ;             bit 6&7 -> luminosity base level (0,1,2,3)
3025                                    <1>     ;             bit 4&5 -> blue level (0,1,2,3)
3026                                    <1>     ;             bit 2%3 -> green level (0,1,2,3)
3027                                    <1>     ;             bit 0&1 -> red level (0,1,2,3)
3028                                    <1>     ;           Example: total red level : luminosity + red level
3029                                    <1>     ;           Luminosity base level: 0 -> 16
3030                                    <1>     ;                                  1 -> 32
3031                                    <1>     ;                                  2 -> 64
3032                                    <1>     ;                                  3 -> 128
3033                                    <1>     ;           Color level:
3034                                    <1>     ;                                 0 -> 0
3035                                    <1>     ;                                 1 -> luminosity level
3036                                    <1>     ;                                 2 -> luminosity level + 64
3037                                    <1>     ;                                 3 -> 255
3038                                    <1>     ;          Luminosity base level = min (R,G,B)
3039                                    <1>     ;                  if it is <16, it will be set to 16
3040                                    <1>     ;          Color levels: Color values are fixed to (nearest)
3041                                    <1>     ;              one of all possible set level (step) values
3042                                    <1>     ;                 (according to luminosity base level); then
3043                                    <1>     ;                 color levels are set to R-L, G-L, B-L.
3044                                    <1>     ;       For example: If luminosity base level is 32
3045                                    <1>     ;                 all possible set values are 0, 32, 96, 255.
3046                                    <1>     ;
3047                                    <1>     ;          * RGB (TRUE) colors to 16 colors conversion:
3048                                    <1>     ;          16 colors: R,B,G,L bits (4 bits)
3049                                    <1>     ;                If any one of R,G,B >= 128 L = 1
3050                                    <1>     ;                If max. value of (R,G,B) >= 32, it is 1
3051                                    <1>     ;                   else all color bits (R&G&B&L) are 0
3052                                    <1>     ;                If the second value >= max. value / 2
3053                                    <1>     ;                   it is 1
3054                                    <1>     ;                If third value value >= max. value / 2
3055                                    <1>     ;                   it is 1
3056                                    <1>     ;          Example: R = 132, G = 64, B = 78
3057                                    <1>     ;                  L = 1, R = 1
3058                                    <1>     ;                  G < 66 --> G = 0
3059                                    <1>     ;                  B >= 66 --> B = 1
3060                                    <1>     ;
3061                                    <1>     ; 10/12/2020
3062                                    <1>     ; SET VIDEO MODE (& RETURN LFB INFO for VESA VBE VIDEO MODES)
3063                                    <1>     ;
3064                                    <1>     ;     BH = 8 = Set Video Mode
3065                                    <1>     ;
3066                                    <1>     ;            BL = Requested Video Mode (method)
3067                                    <1>     ;            If BL = 0FFh
3068                                    <1>     ;               CX = VESA VBE Video Mode
3069                                    <1>     ;               ; 11/12/2020
3070                                    <1>     ;               If EDX > 0 -> LFB INFO (user) buffer addr
3071                                    <1>     ;            If BL < 0FFh, it is VGA/CGA video mode and
3072                                    <1>     ;                  CX & EDX will not be used
3073                                    <1>     ;
3074                                    <1>     ;          NOTE: The last VESA VBE video mode is 11Bh but
3075                                    <1>     ;          TRDOS 386 will permit to set video mode upto 11Fh.
3076                                    <1>     ;          Above 11Fh, from 140h to 1FEh, it will be accepted
3077                                    <1>     ;          as Bochs/Plex86 emulator video mode and it will be
3078                                    <1>     ;          used only if [vbe3] = 2 and detected
3079                                    <1>     ;          video bios is BOCHS/PLEX86/QEMU/VIRTUALBOX vbios.
3080                                    <1>     ;
3081                                    <1>     ; Outputs:
3082                                    <1>     ;     EAX = Requested (Proper) video mode number + 1
3083                                    <1>     ;           ("dec eax" by user will give requested video mode),
3084                                    <1>     ;
3085                                    <1>     ;     If BL input is 0FFh
3086                                    <1>     ;        EDX = LFBINFO table/structure (in user's buffer addr)
3087                                    <1>     ;        EDX = 0 -> Invalid LFBINFO (do not use it)
3088                                    <1>     ;
3089                                    <1>     ;     EAX = 0 -> Error (but EDX will not be changed)
3090                                    <1>     ;
3091                                    <1>     ; 03/12/2020
3092                                    <1>     ; VESA VBE3 VIDEO BIOS (32 bit) PROTECTED MODE INTERFACE SETTINGS
3093                                    <1>     ;
3094                                    <1>     ;     BH = 9 = set/get VBE3 Protected Mode Interface parameters
3095                                    <1>     ;
3096                                    <1>     ;           BL = 0 - Disable protected mode interface
3097                                    <1>     ;                  ([pmi32] = 0)
3098                                    <1>     ;                  Return: AL = 1
3099                                    <1>     ;           BL = 1 - Enable protected mode Interface
3100                                    <1>     ;                  ([pmi32] = 1)
3101                                    <1>     ;                  Return: AL = 2
3102                                    <1>     ;           BL = 2 - Get protected mode interface status
3103                                    <1>     ;                  Return: AL = [pmi32] + 1 (AL = 1 or 2)
3104                                    <1>     ;
3105                                    <1>     ;           If [vbe3] <> 3 --> AL = 0
3106                                    <1>     ;
3107                                    <1>     ;           ; 17/01/2021
3108                                    <1>     ;           BL = 3 - Disable/Cancel restore permission to user
3109                                    <1>     ;                  Return: AL = 1 (if disabled) or 0
3110                                    <1>     ;           BL = 4 - Enable/Give restore permission to user
3111                                    <1>     ;                  Return: AL = 2 (if enabled) or 0
3112                                    <1>     ;           BL = 5 - Get video state save/restore status
3113                                    <1>     ;                   (permission status)
```

```
3114                    <1>     ;                   Return: AL = Status (enabled = 1)
3115                    <1>     ;                        ; 22/01/2021
3116                    <1>     ;                        AH = state options ([srvso])
3117                    <1>     ;            BL = 6 - Return VESA VBE number/status
3118                    <1>     ;                   Return: AX = status
3119                    <1>     ;                       if AH =  2, AL > 0 : Emulator
3120                    <1>     ;                        AH =  3, VESA VBE3 video bios
3121                    <1>     ;
3122                    <1>     ;            BL > 6 : not implemented (17/01/2021)
3123                    <1>     ;
3124                    <1>     ;     ; 19/01/2021 ([u.uid] check)
3125                    <1>     ;     Note: Enabling/Disabling are done by root ([u.uid] = 0)
3126                    <1>     ;           while [multi_tasking] = 0.
3127                    <1>     ;
3128                    <1>     ; 23/12/2020
3129                    <1>     ; VIDEO MEMORY MAPPING:
3130                    <1>     ;     BH = 10 = Map video memory to user's buffer
3131                    <1>     ;
3132                    <1>     ;        BL = 0 : CGA memory (0B8000h) map (32K)
3133                    <1>     ;        BL = 1 : VGA memory (0A0000h) map (64K)
3134                    <1>     ;        BL = 2 : SVGA memory (LFB) map to user's buffer
3135                    <1>     ;
3136                    <1>     ;     ECX = User's buffer addr (low 12 bits will be cleared)
3137                    <1>     ;     EDX = Buffer size in bytes (if BL = 2)
3138                    <1>     ;           (will be trimmed if LFB size < EDX)
3139                    <1>     ; Return:
3140                    <1>       ;   EAX = physical address of video memory (buffer)
3141                    <1>     ;     EBX = mapped (actual) size of video memory (bytes)
3142                    <1>     ;     ECX = virtual start address of user's video buffer
3143                    <1>     ;     EDX is same with EDX input
3144                    <1>     ;
3145                    <1>     ;     (Note: Memory page boundaries will be applied
3146                    <1>     ;      to buffer size and buff start addr by rounding down.
3147                    <1>     ;      Rounded size & address values must not be zero.)
3148                    <1>     ;     -Normally, it is expected to request mapping by using
3149                    <1>     ;      correct buffer size of current or desired video mode-
3150                    <1>     ;
3151                    <1>     ;     EAX = 0 -> error ! memory can not mapped to user
3152                    <1>     ;
3153                    <1>     ; 04/01/2021
3154                    <1>     ; SET/READ COLOR PALETTE (set/read DAC color registers)
3155                    <1>     ;     ((256 colors (8bpp) VGA/CGA video hardware feature))
3156                    <1>     ;
3157                    <1>     ;     BH = 11 = Set/Read DAC color registers
3158                    <1>     ;
3159                    <1>     ;        (B<4 Original method for std VGA video hardware)
3160                    <1>     ;        BL = 0 : Read all DAC color registers (256 colors)
3161                    <1>     ;            (6 bit colors, in RGB order)
3162                    <1>     ;        BL = 1 : Set all DAC color registers (256 colors)
3163                    <1>     ;            (6 bit colors, in RGB order)
3164                    <1>     ;        BL = 2 : Read single DAC color register
3165                    <1>     ;            (6 bit color, in RGB order)
3166                    <1>     ;           ((EAX will return with color value))
3167                    <1>     ;           CL = DAC color register (index)
3168                    <1>     ;        BL = 3 : Set/Write single DAC color register
3169                    <1>     ;            (6 bit color, in RGB order, bit 6&7 are 0)
3170                    <1>     ;           ECX byte 0 - DAC color register
3171                    <1>     ;           ECX byte 1 - Red (6 bit)
3172                    <1>     ;           ECX byte 2 - Green (6 bit)
3173                    <1>     ;           ECX byte 3 - Blue (6 bit)
3174                    <1>     ;        (BL>3 Alternative method for BMP files etc.)
3175                    <1>     ;        BL = 4 : Read all DAC color registers (256 colors)
3176                    <1>     ;            (8 bit colors, in BGR order, bit 0&1 is 0)
3177                    <1>     ;        BL = 5 : Set all DAC color registers (256 colors)
3178                    <1>     ;            (8 bit colors, in BGR order, bit 0&1 is 0)
3179                    <1>     ;        BL = 6 : Read single DAC color register
3180                    <1>     ;            (8 bit color, in BGR order, bit 0&1 is 0)
3181                    <1>     ;           ((EAX will return with color value))
3182                    <1>     ;           CL = DAC color register (index)
3183                    <1>     ;        BL = 7 : Set/Write single DAC color register
3184                    <1>     ;            (8 bit color, bit 0&1 are 0)
3185                    <1>     ;           ECX byte 0 - DAC color register
3186                    <1>     ;           ECX byte 1 - Blue (8 bit)
3187                    <1>     ;           ECX byte 2 - Green (8 bit)
3188                    <1>     ;           ECX byte 3 - Red (8 bit)
3189                    <1>     ;
3190                    <1>     ;       BL > 7 : invalid (not implemented)
3191                    <1>     ;
3192                    <1>     ;   if BL bit 2 is 1, 6 bit colors converted to 8 bit colors
3193                    <1>     ;      (low two bits of color bytes will be 0)
3194                    <1>     ;     ((color byte 00111111b will be converted to 11111100b))
3195                    <1>     ;     and RGB byte order will be
3196                    <1>     ;           byte 0 - Blue (low 2 bits ares 0)
3197                    <1>     ;           byte 1 - Green (low 2 bits are 0)
3198                    <1>     ;           byte 2 - Red (low 2 bits are 0)
3199                    <1>     ;           byte 3 - pad (or zero byte)
3200                    <1>     ;     and 256 colors buffer size must be 256*4 = 1024 bytes
3201                    <1>     ;   if BL bit 2 is 0, 6 bit colors will be used directly
3202                    <1>     ;       (high two bits of 8 bit color bytes will be 0)
3203                    <1>     ;     ((dac color 111111b will be converted to 00111111b))
3204                    <1>     ;           byte 0 - Red (high 2 bits are 0)
3205                    <1>     ;           byte 1 - Green (high 2 bits are 0)
3206                    <1>     ;           byte 2 - Blue (high 2 bits are 0)
3207                    <1>     ;     and 256 colors buffer size must be 256*3 = 768 bytes
3208                    <1>     ;
3209                    <1>     ;     ECX = User's buffer addr (256*3 = 768 bytes) or
3210                    <1>     ;           Color
3211                    <1>     ;
3212                    <1>     ; Return:
3213                    <1>       ;   EAX = buffer size (for BL input = 0,1,4,5)
3214                    <1>     ;           or color value (for BL input = 2,3,6,7)
3215                    <1>     ;
3216                    <1>     ; 10/01/2021
3217                    <1>     ; SET/READ FONT DATA
3218                    <1>     ;
```

```
3219                    <1>     ;      BH = 12 = Set/Read Character Font Data
3220                    <1>     ;
3221                    <1>     ;         BL = 0 : Disable system font overwrite
3222                    <1>     ;         BL = 1 : Enable system font overwrite
3223                    <1>     ;         BL = 2 : Read system font 8x8
3224                    <1>     ;         BL = 3 : Read system font 8x14
3225                    <1>     ;         BL = 4 : Read system font 8x16
3226                    <1>     ;         BL = 5 : Read user defined font 8x8
3227                    <1>     ;         BL = 6 : Read user defined font 8x16
3228                    <1>     ;         BL = 7 : Write system font 8x8
3229                    <1>     ;         BL = 8 : Write system font 8x14
3230                    <1>     ;         BL = 9 : Write system font 8x16
3231                    <1>     ;         BL = 10 : Write user defined font 8x8
3232                    <1>     ;         BL = 11 : Write user defined font 8x16
3233                    <1>     ;
3234                    <1>     ;        BL > 11 : invalid (not implemented)
3235                    <1>     ;
3236                    <1>     ;      For BL = 1 to 11
3237                    <1>     ;         ECX = number of characters (>= 256)
3238                    <1>     ;         EDX = first character (ascii code in DL)
3239                    <1>     ;         ESI = user's buffer address
3240                    <1>     ;
3241                    <1>     ; Return:
3242                    <1>       ;    EAX = number of characters (ecx input)
3243                    <1>     ;      EAX = 0 -> error
3244                    <1>     ;      (EAX = 256 for BL = 0 and 1 if successful)
3245                    <1>     ;
3246                    <1>     ; Note: system font overwrite permission will be
3247                    <1>     ;       given if [multi_tasking] = 0
3248                    <1>     ;      and [u.uid] = 0 (BL = 1) ; 19/01/2021
3249                    <1>     ;        and if [ufont] bit 7 is 1 (BL = 7,8,9)
3250                    <1>     ;
3251                    <1>     ; 18/01/2021
3252                    <1>     ; SAVE/RESTORE STANDARD VGA VIDEO STATE
3253                    <1>     ;
3254                    <1>     ;      BH = 13 = Save/Restore std VGA video state
3255                    <1>     ;
3256                    <1>     ;         BL = 0 : Save VGA state (without DAC regs)
3257                    <1>     ;             Return: EAX = VideoStateID (>0)
3258                    <1>     ;                     EAX = 0 (failed!)
3259                    <1>     ;              (size: 110 bytes for TRDOS 386 v2.0.3)
3260                    <1>     ;         BL = 1 : Restore VGA state (without DAC regs)
3261                    <1>     ;             ECX = VideoStateID (to be verified)
3262                    <1>     ;               Return: EAX = Restore size (>0)
3263                    <1>     ;         BL = 2 : Save VGA state (complete)
3264                    <1>     ;             Return: EAX = VideoStateID (>0)
3265                    <1>     ;                     EAX = 0 (failed!)
3266                    <1>     ;              (size: 882 bytes for TRDOS 386 v2.0.3)
3267                    <1>     ;         BL = 3 : Restore VGA state (complete)
3268                    <1>     ;             ECX = VideoStateID (to be verified)
3269                    <1>     ;               Return: EAX = Restore size (>0)
3270                    <1>     ;
3271                    <1>     ;      * Above options are for saving
3272                    <1>     ;      *       video state to system memory
3273                    <1>     ;      *      (location: VBE3VIDEOSTATE, 2048 bytes)
3274                    <1>     ;
3275                    <1>     ;         BL = 4 : Save VGA state (without DAC regs)
3276                    <1>     ;             ECX = buffer address
3277                    <1>     ;             Return: EAX = transfer count
3278                    <1>     ;              (size: 110 bytes for TRDOS 386 v2.0.3)
3279                    <1>     ;             ECX = buffer address
3280                    <1>     ;         BL = 5 : Restore VGA state (without DAC regs)
3281                    <1>     ;             ECX = buffer address
3282                    <1>     ;               Return: EAX = transfer count
3283                    <1>     ;         BL = 6 : Save VGA state (complete)
3284                    <1>     ;             ECX = buffer address
3285                    <1>     ;             Return: EAX = transfer count
3286                    <1>     ;              (size: 882 bytes for TRDOS 386 v2.0.3)
3287                    <1>     ;         BL = 7 : Restore VGA state (complete)
3288                    <1>     ;             ECX = buffer address
3289                    <1>     ;               Return: EAX = transfer count
3290                    <1>     ;
3291                    <1>     ;      * Above options are for saving
3292                    <1>     ;      *       video state to user's buffer
3293                    <1>     ;      *      (buffer size: 110 bytes or 882 bytes)
3294                    <1>     ;
3295                    <1>     ;        BL > 7 : invalid (not implemented)
3296                    <1>     ;
3297                    <1>     ; 18/01/2021
3298                    <1>     ; SAVE/RESTORE SUPER VGA (VESA VBE 2/3) VIDEO STATE
3299                    <1>     ;
3300                    <1>     ;      BH = 14 = Save/Restore SVGA video state
3301                    <1>     ;
3302                    <1>     ;         BL = options
3303                    <1>     ;             bit 0 - Save (0) or Restore (1)
3304                    <1>     ;             bit 1 - controller hardware state
3305                    <1>     ;             bit 2 - BIOS data state
3306                    <1>     ;             bit 3 - DAC state
3307                    <1>     ;             bit 4 - (extended) Register state
3308                    <1>     ;             bit 5 - system (0) or user (1) memory
3309                    <1>     ;             bit 6 - verify without transfer
3310                    <1>     ;             bit 7 - not used (must be 0)
3311                    <1>     ;
3312                    <1>     ;           if bit 0 = 0 and bit 5 = 0
3313                    <1>     ;             Return:      EAX = VideoStateID (>0)
3314                    <1>     ;           if bit 0 = 1
3315                    <1>     ;             ECX = VideoStateID (bit 5 = 0)
3316                    <1>     ;             Return: EAX = restore (transfer) size
3317                    <1>     ;           if bit 5 = 1
3318                    <1>     ;             ECX = Buffer address
3319                    <1>     ;             Return: EAX = transfer count (size)
3320                    <1>     ;
3321                    <1>     ;           ECX = Buffer address or VideoStateID
3322                    <1>     ;
3323                    <1>     ;        BL > 127 : invalid (not implemented)
```

```
3324                                    <1>       ;
3325                                    <1>       ;  Note: Required buffer size may be > 2048 bytes
3326                                    <1>       ;       (function fails when buff size > 2048 bytes)
3327                                    <1>       ;       proper option must be used
3328                                    <1>       ;
3329                                    <1>       ; 18/01/2021
3330                                    <1>       ; READ VESA EDID (EXTENDED DISPLAY IDENTIFICATION DATA)
3331                                    <1>       ;
3332                                    <1>       ;      BH = 15 = Read VESA EDID for connected monitor
3333                                    <1>       ;               (copy EDID to user)
3334                                    <1>       ;
3335                                    <1>       ;         BL = any
3336                                    <1>       ;
3337                                    <1>       ;      Input:
3338                                    <1>       ;          ECX = user's (EDID) buffer address
3339                                    <1>       ;              (buffer size: 128 bytes)
3340                                    <1>       ;      Output:
3341                                    <1>       ;          EAX = 128 (EDID size)
3342                                    <1>       ;          or EAX = 0 -> Error!
3343                                    <1>       ;             (EDID not ready or buffer addr error)
3344                                    <1>       ;
3345                                    <1>
3346                                    <1>       ; 16/05/2016
3347 0000E2E2 31C0                     <1>       xor    eax, eax
3348 0000E2E4 A3[64030300]             <1>       mov    [u.r0], eax
3349 0000E2E9 20FF                     <1>       and    bh, bh
3350 0000E2EB 0F8586020000             <1>       jnz    sysvideo_13 ; 11/07/2016
3351                                    <1>
3352                                    <1>       ;; 21/11/2020 (TRDOS 386 v2.0.3)
3353                                    <1>       ;; tty/text mode transfers are only for video mode 3
3354                                    <1>
3355                                    <1>       ; 22/11/2020
3356                                    <1>       ;cmp   byte [CRT_MODE], 3 ; 80x25 text, 16 colors
3357                                    <1>       ;jne   sysret ; invalid (nothing to do), [u.r0] = 0
3358                                    <1>
3359                                    <1>       ; 23/11/2020
3360                                    <1>       ; bit 7,6,5,4 of BL are reserved and it must be 0
3361                                    <1>       ;             for current 'sysvideo' version
3362 0000E2F1 F6C3F0                   <1>       test   bl, 0F0h
3363 0000E2F4 0F85F2F5FFFF             <1>       jnz    sysret ; invalid (undefined) !
3364                                    <1>
3365                                    <1>       ; Video mode 0, 80*25 text mode, CGA 16 colors  ; [CRT_MODE] = 3
3366 0000E2FA 88DF                     <1>       mov    bh, bl
3367 0000E2FC C0EF02                   <1>       shr    bh, 2 ; 4..7 -> 1, 8..11 -> 2, 12..15 -> 3
3368                                    <1>       ; 21/11/2020
3369                                    <1>       ;and   bh, bh
3370 0000E2FF 0F8592000000             <1>         jnz  sysvideo_4  ; Display page window transfer etc.
3371                                    <1>
3372                                    <1>       ; Display page (complete) transfer
3373                                    <1>
3374 0000E305 BF00800B00               <1>       mov    edi, 0B8000h
3375                                    <1>       ; dl = display page number, destination
3376 0000E30A 66B80010                 <1>       mov    ax, 4096 ; 21/11/2020
3377 0000E30E 20D2                     <1>       and    dl, dl
3378 0000E310 7413                     <1>       jz     short sysvideo_1
3379 0000E312 80FA07                   <1>       cmp    dl, 7
3380 0000E315 0F87D1F5FFFF             <1>       ja     sysret ; invalid (nothing to do), [u.r0] = 0
3381                                    <1> sysvideo_0:
3382                                    <1>       ; page lenght = 4096 bytes (but page content is 80*25*2 bytes)
3383 0000E31B 81C700100000             <1>       add    edi, 4096 ; 21//11/2020 ([CRT_LEN] = 1000h for mode 3)
3384 0000E321 FECA                     <1>       dec    dl
3385 0000E323 75F6                     <1>       jnz    short sysvideo_0
3386                                    <1> sysvideo_1:
3387 0000E325 80E303                   <1>       and    bl, 3
3388 0000E328 752A                     <1>       jnz    short sysvideo_2 ; user to system display page transfer
3389                                    <1>       ; cl = display page number, source
3390 0000E32A 80F907                   <1>       cmp    cl, 7
3391 0000E32D 0F87B9F5FFFF             <1>       ja     sysret ; invalid (nothing to do), [u.r0] = 0
3392                                    <1>       ; system to system video/display page transfer (mode 0)
3393                                    <1>       ; 21/11/2020
3394                                    <1>       ;mov   esi, 0B8000h
3395                                    <1>       ;movzx eax, cl
3396                                    <1>       ;mov   edx, 4096 ; [CRT_LEN] = 1000h for video mode 3
3397                                    <1>       ;mov   ecx, edx
3398                                    <1>       ;mul   edx
3399                                    <1>       ;add   esi, eax
3400 0000E333 0FB6F1                   <1>       movzx  esi, cl
3401 0000E336 66C1E60C                 <1>       shl    si, 12 ; * 4096
3402 0000E33A 81C600800B00             <1>       add    esi, 0B8000h
3403                                    <1>
3404                                    <1>       ; 21/11/2020
3405                                    <1>       ;mov   ecx, 4096
3406 0000E340 89C1                     <1>       mov    ecx, eax ; 4096
3407                                    <1>       ;mov   [u.r0], ecx ; 4096 bytes
3408 0000E342 66890D[64030300]         <1>       mov    [u.r0], cx
3409                                    <1>
3410 0000E349 66C1E902                 <1>       shr    cx, 2 ; / 4
3411 0000E34D F3A5                     <1>       rep    movsd
3412 0000E34F E998F5FFFF               <1>       jmp    sysret
3413                                    <1> sysvideo_2:
3414 0000E354 80FB02                   <1>       cmp    bl, 2
3415 0000E357 0F878FF5FFFF             <1>         ja      sysret; invalid (user to user), [u.r0] = 0
3416 0000E35D 721D                     <1>       jb     short sysvideo_3 ; user to system
3417                                    <1>       ; system to user video/display page transfer (mode 0)
3418 0000E35F 89FE                     <1>       mov    esi, edi
3419 0000E361 89CF                     <1>       mov    edi, ecx ; user buffer
3420                                    <1>       ; 21/11/2020
3421 0000E363 89C1                     <1>       mov    ecx, eax ; 4096
3422 0000E365 E85A2F0000               <1>       call   transfer_to_user_buffer ; fast transfer
3423 0000E36A 0F827CF5FFFF             <1>       jc     sysret ; [u.r0] = 0
3424                                    <1>       ; 21/11/2020
3425                                    <1>       ;mov   [u.r0], ecx
3426 0000E370 66890D[64030300]         <1>       mov    [u.r0], cx
3427 0000E377 E970F5FFFF               <1>       jmp    sysret
3428                                    <1> sysvideo_3:
```

```
3429                                    <1>       ; user to system video/display page transfer (mode 0)
3430 0000E37C 89CE                      <1>       mov    esi, ecx ; user buffer
3431                                    <1>       ; edi = video page address
3432                                    <1>       ; 21/11/2020
3433 0000E37E 89C1                      <1>       mov    ecx, eax ; 4096
3434 0000E380 E8892F0000                <1>       call   transfer_from_user_buffer ; fast transfer
3435 0000E385 0F8261F5FFFF              <1>       jc     sysret ; [u.r0] = 0
3436                                    <1>       ; 21/11/2020
3437                                    <1>       ;mov   [u.r0], ecx
3438 0000E38B 66890D[64030300]          <1>       mov    [u.r0], cx
3439 0000E392 E955F5FFFF                <1>       jmp    sysret
3440                                    <1>
3441                                    <1> sysvideo_4:
3442                                    <1>       ; 23/11/2020 (TRDOS 386 v2.0.3)
3443                                    <1>
3444                                    <1>       ; Display page window transfer etc.
3445 0000E397 80E303                    <1>       and    bl, 3
3446 0000E39A 0F85F7000000              <1>         jnz     sysvideo_9 ; user to system or system to user
3447                                    <1>       ; 21/11/2020
3448                                    <1>       ; system to system video/display page window transfer (mode 0)
3449 0000E3A0 81FE00800B00              <1>       cmp    esi, 0B8000h ; source buffer address (system)
3450 0000E3A6 0F8240F5FFFF              <1>       jb     sysret
3451 0000E3AC 81FE00000C00              <1>       cmp    esi, 0B8000h+(4096*8)
3452 0000E3B2 0F8334F5FFFF              <1>       jnb    sysret
3453 0000E3B8 81FF00800B00              <1>       cmp    edi, 0B8000h ; destination buffer address (system)
3454 0000E3BE 0F8228F5FFFF              <1>       jb     sysret
3455 0000E3C4 81FF00000C00              <1>         cmp     edi, 0B8000h+(4096*8)
3456 0000E3CA 0F831CF5FFFF              <1>       jnb    sysret
3457                                    <1>       ;
3458 0000E3D0 51                        <1>       push   ecx ; X1 and Y1 position - top left column, row
3459 0000E3D1 0FB7C1                    <1>       movzx  eax, cx ; top left column
3460                                    <1>       ; 21/11/2020
3461 0000E3D4 C1E910                    <1>       shr    ecx, 16 ; top row
3462 0000E3D7 740E                      <1>       jz     short sysvideo_4_0 ; bypass following code
3463 0000E3D9 52                        <1>       push   edx ; X2 and Y2 position - bottom right column, row
3464 0000E3DA 50                        <1>       push   eax
3465 0000E3DB 66B8A000                  <1>       mov    ax, 80*2 ; 80 colums, 160 bytes per row
3466 0000E3DF F7E1                      <1>       mul    ecx
3467                                    <1>              ; eax = offset for start row number
3468 0000E3E1 01C6                      <1>       add    esi, eax
3469 0000E3E3 01C7                      <1>       add    edi, eax
3470 0000E3E5 58                        <1>       pop    eax
3471 0000E3E6 5A                        <1>       pop    edx
3472                                    <1> sysvideo_4_0:
3473 0000E3E7 66D1E0                    <1>       shl    ax, 1 ; * 2 ; convert start column number to offset
3474 0000E3EA 7404                      <1>       jz     short sysvideo_4_1
3475 0000E3EC 01C6                      <1>       add    esi, eax
3476 0000E3EE 01C7                      <1>       add    edi, eax
3477                                    <1>              ; esi = source page window start offset
3478                                    <1>              ; edi = destination page window start offset
3479                                    <1> sysvideo_4_1:
3480 0000E3F0 59                        <1>       pop    ecx
3481                                    <1>       ;mov   eax, 0B8000h+(80*25*2*8)
3482                                    <1>       ; 21/11/2020
3483 0000E3F1 B800000C00                <1>       mov    eax, 0B8000h+(4096*8)
3484 0000E3F6 39C6                      <1>       cmp    esi, eax
3485 0000E3F8 0F83EEF4FFFF              <1>       jnb    sysret ; out of video page
3486 0000E3FE 39C6                      <1>       cmp    esi, eax
3487 0000E400 0F83E6F4FFFF              <1>       jnb    sysret  ;out of video page
3488                                    <1>
3489 0000E406 56                        <1>       push   esi ; ****
3490 0000E407 57                        <1>       push   edi ; ***
3491 0000E408 52                        <1>       push   edx ; **
3492 0000E409 51                        <1>       push   ecx ; *
3493 0000E40A C1E910                    <1>       shr    ecx, 16 ; top row
3494 0000E40D C1EA10                    <1>       shr    edx, 16 ; bottom row
3495                                    <1>       ; 21/11/2020
3496                                    <1>       ;cmp   ecx, 24 ; max. 25 rows
3497 0000E410 6683F918                  <1>       cmp    cx, 24
3498 0000E414 7778                      <1>       ja     short sysvideo_6 ; invalid, [u.r0] = 0
3499                                    <1>       ;cmp   edx, 24 ; max. 25 rows
3500 0000E416 6683FA18                  <1>       cmp    dx, 24
3501 0000E41A 7772                      <1>       ja     short sysvideo_6 ; invalid, [u.r0] = 0
3502 0000E41C 28CA                      <1>       sub    dl, cl ; end >= start
3503 0000E41E 726E                      <1>       jc     short sysvideo_6 ; invalid, [u.r0] = 0
3504                                    <1>       ; 21/11/2020
3505 0000E420 89D3                      <1>       mov    ebx, edx ; row count - 1
3506 0000E422 7415                      <1>       jz     short sysvideo_4_2
3507 0000E424 50                        <1>       push   eax ; *****
3508 0000E425 B8A0000000                <1>       mov    eax, 80*2 ; bytes per row
3509 0000E42A F7E3                      <1>       mul    ebx ; 21/11/2020
3510                                    <1>              ; eax = window end offset
3511                                    <1>              ; (for the last row, before adding column bytes)
3512 0000E42C 01C6                      <1>       add    esi, eax
3513 0000E42E 01C7                      <1>       add    edi, eax
3514 0000E430 58                        <1>       pop    eax ; ***** ; mode 3 video memory end (0C000h)
3515 0000E431 39C6                      <1>       cmp    esi, eax
3516                                    <1>       ;ja    short sysvideo_6 ; invalid, [u.r0] = 0
3517 0000E433 7359                      <1>       jnb    short sysvideo_6 ; 21/11/2020
3518 0000E435 39C7                      <1>       cmp    edi, eax
3519                                    <1>       ;ja    short sysvideo_6 ; invalid, [u.r0] = 0
3520 0000E437 7355                      <1>       jnb    short sysvideo_6 ; 21/11/2020
3521                                    <1> sysvideo_4_2:
3522 0000E439 59                        <1>       pop    ecx ; *
3523 0000E43A 5A                        <1>       pop    edx ; **
3524 0000E43B 81E1FFFF0000              <1>       and    ecx, 0FFFFh
3525 0000E441 81E2FFFF0000              <1>       and    edx, 0FFFFh
3526                                    <1>       ; 21/11/2020
3527                                    <1>       ;cmp   ecx, 79 ; max. 80 columns
3528 0000E447 6683F94F                  <1>       cmp    cx, 79
3529 0000E44B 7743                      <1>       ja     short sysvideo_7 ; invalid, [u.r0] = 0
3530                                    <1>       ;cmp   edx, 79 ; max. 80 columns
3531 0000E44D 6683FA4F                  <1>       cmp    dx, 79
3532 0000E451 773D                      <1>       ja     short sysvideo_7 ; invalid, [u.r0] = 0
3533 0000E453 28CA                      <1>       sub    dl, cl
```

```
3534 0000E455 7639              <1>       jna    short sysvideo_7 ; invalid, [u.r0] = 0
3535                            <1>       ; 21/11/2020
3536 0000E457 740E              <1>       jz     short sysvideo_4_3
3537                            <1>       ; edx = column count (width) - 1
3538 0000E459 D0E2              <1>       shl    dl, 1 ; * 2 ; byte offset (in end row)
3539 0000E45B 01D6              <1>       add    esi, edx
3540 0000E45D 01D7              <1>       add    edi, edx
3541                            <1>              ; esi = source page window end offset
3542                            <1>              ; edi = destination page window end offset
3543 0000E45F 39C6              <1>       cmp    esi, eax ; video memory end
3544                            <1>       ;ja    short sysvideo_7
3545 0000E461 732D              <1>       jnb    short sysvideo_7 ; 21/11/2020
3546 0000E463 39C7              <1>       cmp    edi, eax ; video memory end
3547                            <1>       ;ja    short sysvideo_7
3548 0000E465 7329              <1>       jnb    short sysvideo_7 ; 21/11/2020
3549                            <1> sysvideo_4_3:
3550 0000E467 5F                <1>       pop    edi ; ***
3551 0000E468 5E                <1>       pop    esi ; ****
3552 0000E469 FEC3              <1>       inc    bl ; row count - 1 -> row count
3553 0000E46B FEC2              <1>       inc    dl ; column count
3554 0000E46D 88D7              <1>       mov    bh, dl
3555 0000E46F D0E2              <1>       shl    dl, 1 ; convert column count to byte offset
3556                            <1>       ; 21/11/2020
3557                            <1>       ; esi = source page window start offset
3558                            <1>       ; edi = destination page window start offset
3559 0000E471 B8A0000000        <1>       mov    eax, 80*2 ; bytes per row
3560                            <1>       ; Note: 160 bytes per row (even if move count < 160)
3561                            <1> sysvideo_5:
3562 0000E476 88F9              <1>       mov    cl, bh ; move/transfer -word- count per row
3563 0000E478 0115[64030300]    <1>       add    [u.r0], edx ; transfer count in bytes
3564 0000E47E F366A5            <1>       rep    movsw
3565 0000E481 01C6              <1>       add    esi, eax ; + 160 bytes to next row
3566 0000E483 01C7              <1>       add    edi, eax ; + 160 bytes to next row
3567 0000E485 FECB              <1>       dec    bl ; remain count of rows
3568 0000E487 75ED              <1>       jnz    short sysvideo_5
3569 0000E489 E95EF4FFFF        <1>       jmp    sysret
3570                            <1>
3571                            <1> sysvideo_6:
3572 0000E48E 59                <1>       pop    ecx ; *
3573 0000E48F 5A                <1>       pop    edx ; **
3574                            <1> sysvideo_7:
3575 0000E490 5F                <1>       pop    edi ; ***
3576 0000E491 5E                <1>       pop    esi ; ****
3577                            <1> sysvideo_8:
3578 0000E492 E955F4FFFF        <1>       jmp    sysret
3579                            <1>
3580                            <1> sysvideo_9:
3581                            <1>       ; user to system or system to user window transfer
3582 0000E497 80FB02            <1>       cmp    bl, 2
3583                            <1>       ;ja    sysret ; user to user transfer is invalid
3584                            <1>                     ; [u.r0] = 0
3585 0000E49A 77F6              <1>       ja     short sysvideo_8 ; 26/12/2020
3586                            <1>
3587 0000E49C 56                <1>       push   esi ; ****
3588 0000E49D 57                <1>       push   edi ; ***
3589 0000E49E 52                <1>       push   edx ; **
3590 0000E49F 51                <1>       push   ecx ; *
3591                            <1>
3592 0000E4A0 C1E910            <1>       shr    ecx, 16 ; top row
3593 0000E4A3 C1EA10            <1>       shr    edx, 16 ; bottom row
3594                            <1>
3595                            <1>       ; 21/11/2020
3596                            <1>       ;cmp   ecx, 24 ; max. 25 rows
3597 0000E4A6 6683F918          <1>       cmp    cx, 24
3598 0000E4AA 77E2              <1>       ja     short sysvideo_6 ; invalid, [u.r0] = 0
3599                            <1>       ;cmp   edx, 24 ; max. 25 rows
3600 0000E4AC 6683FA18          <1>       cmp    dx, 24
3601 0000E4B0 77DC              <1>       ja     short sysvideo_6 ; invalid, [u.r0] = 0
3602 0000E4B2 28CA              <1>       sub    dl, cl
3603 0000E4B4 72D8              <1>       jc     short sysvideo_6 ; invalid, [u.r0] = 0
3604                            <1>
3605                            <1>       ;mov   ch, cl ; top row
3606                            <1>       ;mov   cl, [ACTIVE_PAGE]
3607                            <1>
3608                            <1>       ;mov   edi, 80*25*2 ; 4000
3609                            <1>       ; 21/11/2020
3610                            <1>       ;mov   edi, 4096 ; [CRT_LEN = 4096 for video mode 3
3611                            <1>       ;shl   edi, cl  ; ! wrong for page 2 to page 7 !
3612                            <1>       ;;add  edi, 0B8000h - 80*25*2
3613                            <1>       ;add   edi, 0B8000h - 1000h ; - 4096
3614                            <1>
3615                            <1>       ; 21/11/2020
3616                            <1>       ;xor   eax, eax
3617                            <1>       ;mov   edi, 0B8000h
3618                            <1>       ;and   cl, cl ; is video page = 0 ?
3619                            <1>       ;jz    short sysvideo_9_1 ; yes
3620                            <1>       ; eax = 0
3621                            <1>
3622                            <1> ;sysvideo_9_0:
3623                            <1>       ;add   ax, 4096
3624                            <1>       ;dec   cl
3625                            <1>       ;jnz   short sysvideo_9_0
3626                            <1>       ;add   edi, eax
3627                            <1>       ;      ; edi = video page start address
3628                            <1>
3629                            <1>       ; 21/11/2020
3630 0000E4B6 BF00800B00        <1>       mov    edi, 0B8000h
3631 0000E4BB 803D[BE810100]00  <1>       cmp    byte [ACTIVE_PAGE], 0
3632 0000E4C2 760C              <1>       jna    short sysvideo_9_1
3633                            <1> stsvideo_9_0:
3634 0000E4C4 B010              <1>       mov    al, 16 ; 4096/256
3635 0000E4C6 F625[BE810100]    <1>       mul    byte [ACTIVE_PAGE]
3636 0000E4CC 86E0              <1>       xchg   ah, al ; * 256
3637 0000E4CE 01C7              <1>       add    edi, eax
3638                            <1>              ; edi = video page start address
```

```
3639                                    <1> sysvideo_9_1:
3640                                    <1>       ; bl = transfer direction
3641                                    <1>       ;      ( 1 = from user, 2 = to user)
3642 0000E4D0 88D7                     <1>       mov    bh, dl ; row count - 1
3643                                    <1>       ;mov   dl, ch ; top row
3644                                    <1>       ; 21/11/2020
3645 0000E4D2 08C9                     <1>       or     cl, cl ; top row number
3646 0000E4D4 7408                     <1>       jz     short sysvideo_9_2
3647                                    <1>
3648                                    <1>       ;mov   eax, 80*2
3649 0000E4D6 66B8A000                 <1>       mov    ax, 80*2 ; 160, bytes per row
3650 0000E4DA F7E1                     <1>       mul    ecx ; 22/11/2020
3651 0000E4DC 01C7                     <1>       add    edi, eax
3652                                    <1>              ; edi = window start address for top row
3653                                    <1> sysvideo_9_2:
3654 0000E4DE 59                       <1>       pop    ecx ; *
3655 0000E4DF 5A                       <1>       pop    edx ; **
3656 0000E4E0 81E1FFFF0000             <1>       and    ecx, 0FFFFh
3657 0000E4E6 81E2FFFF0000             <1>       and    edx, 0FFFFh
3658                                    <1>       ; 21/11/2020
3659                                    <1>       ;cmp   ecx, 79 ; max. 80 columns
3660 0000E4EC 6683F94F                 <1>       cmp    cx, 79
3661 0000E4F0 779E                     <1>       ja     short sysvideo_7 ; invalid, [u.r0] = 0
3662                                    <1>       ;cmp   edx, 79 ; max. 80 columns
3663 0000E4F2 6683FA4F                 <1>       cmp    dx, 79
3664 0000E4F6 7798                     <1>       ja     short sysvideo_7 ; invalid, [u.r0] = 0
3665                                    <1>
3666 0000E4F8 28CA                     <1>       sub    dl, cl
3667 0000E4FA 7294                     <1>       jc     short sysvideo_7 ; invalid, [u.r0] = 0
3668                                    <1>
3669 0000E4FC 08C9                     <1>       or     cl, cl ; left column
3670 0000E4FE 7404                     <1>       jz     short sysvideo_9_3 ; 0
3671                                    <1>
3672                                    <1>       ; 21/11/2020
3673 0000E500 D0E1                     <1>       shl    cl, 1  ; column * 2
3674 0000E502 01CF                     <1>       add    edi, ecx
3675                                    <1>              ; edi = window start addr for top left column
3676                                    <1> sysvideo_9_3:
3677 0000E504 88D1                     <1>       mov    cl, dl
3678 0000E506 FEC1                     <1>       inc    cl ; column count
3679 0000E508 D0E1                     <1>       shl    cl, 1 ; column count * 2
3680                                    <1>              ; ecx = tranfer count per row
3681                                    <1>
3682 0000E50A 58                       <1>       pop    eax ; *** (swap address)
3683 0000E50B 5E                       <1>       pop    esi ; ****
3684                                    <1>
3685 0000E50C FEC7                     <1>       inc    bh  ; row count
3686                                    <1>
3687                                    <1>       ;mov   edx, 80*2
3688 0000E50E B2A0                     <1>       mov    dl, 80*2  ; bytes per row
3689                                    <1>       ;
3690 0000E510 80FB01                   <1>       cmp    bl, 1 ; transfer direction
3691 0000E513 7742                     <1>       ja     short sysvideo_11 ; system to user transfer
3692                                    <1>
3693                                    <1>       ; user to system video/display page window transfer (mode 0)
3694 0000E515 21C0                     <1>       and    eax, eax ; swap address
3695 0000E517 7420                     <1>       jz     short sysvideo_10 ; no window swap
3696                                    <1>       ; save previous window content in user's buffer (swap address)
3697 0000E519 56                       <1>       push   esi ; user buffer
3698 0000E51A 57                       <1>       push   edi ; beginning address of the window
3699                                    <1>       ; 21/11/2020
3700 0000E51B 53                       <1>       push   ebx ; save bh
3701 0000E51C 89FE                     <1>       mov    esi, edi
3702 0000E51E 89C7                     <1>       mov    edi, eax
3703                                    <1> sysvideo_9_4:
3704 0000E520 E89F2D0000               <1>       call   transfer_to_user_buffer ; fast transfer
3705 0000E525 7208                     <1>       jc     short sysvideo_9_5
3706                                    <1>       ; ecx = actual transfer count (must be same with input)
3707 0000E527 01D6                     <1>       add    esi, edx ; next row address of (video page) window
3708 0000E529 01CF                     <1>       add    edi, ecx ; next row address of user's window
3709                                    <1>              ; Note: ecx may be less than row length of video page
3710                                    <1>              ; user's window uses offset according to window width
3711 0000E52B FECF                     <1>       dec    bh
3712 0000E52D 75F1                     <1>       jnz    short sysvideo_9_4 ; repeat for next row
3713                                    <1> sysvideo_9_5:
3714 0000E52F 5B                       <1>       pop    ebx ; restore bh
3715 0000E530 5F                       <1>       pop    edi
3716 0000E531 5E                       <1>       pop    esi
3717 0000E532 7305                     <1>       jnc    short sysvideo_10
3718 0000E534 E9B3F3FFFF               <1>       jmp    sysret ; [u.r0] = 0
3719                                    <1> sysvideo_10:
3720                                    <1>       ; user to system video/display page window transfer (mode 0)
3721                                    <1>       ; esi =      user buffer
3722 0000E539 E8D02D0000               <1>       call   transfer_from_user_buffer ; fast transfer
3723 0000E53E 0F82A8F3FFFF             <1>       jc     sysret
3724                                    <1>       ; ecx = actual transfer count (must be same with input)
3725 0000E544 010D[64030300]           <1>       add    [u.r0], ecx ; actual transfer count
3726 0000E54A 01D7                     <1>       add    edi, edx ; next row address of (ivdeo page) window
3727 0000E54C 01CE                     <1>       add    esi, ecx ; next row address of user's window
3728                                    <1>              ; Note: ecx may be less than row length of video page
3729                                    <1>              ; user's window uses offset according to window width
3730 0000E54E FECF                     <1>       dec    bh
3731 0000E550 75E7                     <1>       jnz    short sysvideo_10 ; repeat for next row
3732 0000E552 E995F3FFFF               <1>       jmp    sysret
3733                                    <1>
3734                                    <1> sysvideo_11:
3735                                    <1>       ; system to user video/display page window transfer (mode 0)
3736 0000E557 87FE                     <1>       xchg   edi, esi
3737                                    <1> sysvideo_12:
3738                                    <1>       ; esi = beginning addr of the (screen, video page) window
3739                                    <1>       ; edi =      user's buffer
3740 0000E559 E8662D0000               <1>       call   transfer_to_user_buffer ; fast transfer
3741 0000E55E 0F8288F3FFFF             <1>       jc     sysret
3742                                    <1>       ; ecx = actual transfer count (must be same with input)
3743 0000E564 010D[64030300]           <1>       add    [u.r0], ecx
```

```
3744 0000E56A 01D6              <1>       add    esi, edx ; next row (edx = 160)
3745 0000E56C 01CF              <1>       add    edi, ecx ; next row of the user's window
3746                            <1>                       ; (ecx <= 160)
3747 0000E56E FECF              <1>       dec    bh
3748 0000E570 75E7              <1>       jnz    short sysvideo_12
3749                            <1> sysvideo_12_0:
3750 0000E572 E975F3FFFF        <1>       jmp    sysret
3751                            <1>
3752                            <1> sysvideo_13:
3753                            <1>       ; 28/12/2020
3754 0000E577 80FF01            <1>       cmp    bh, 1
3755 0000E57A 7733              <1>         ja      short sysvideo_15 ; 23/11/2020
3756                            <1>
3757                            <1>       ;cmp   bl, 13
3758                            <1>       ; 02/01/2021
3759 0000E57C 80FB0F            <1>       cmp    bl, 15
3760 0000E57F 77F1              <1>       ja     short sysvideo_12_0 ; invalid (unknown) sub function
3761                            <1>
3762                            <1>       ; 23/11/2020 (TRDOS 386 v2.0.3)
3763                            <1>       ; (major modification, from mode 13h to all VGA modes,
3764                            <1>       ;  except super VGA modes and liner frame buffer method)
3765                            <1>
3766                            <1>       ; BH = 1 = VGA Graphics mode (0A0000h) data transfers
3767                            <1>
3768                            <1>       ; 23/11/2020
3769                            <1>       ; screen width will be saved into EBP register
3770                            <1>
3771 0000E581 88DC              <1>       mov    ah, bl
3772                            <1>       ;and   bl, 0Fh
3773 0000E583 C0EC04            <1>       shr    ah, 4
3774 0000E586 BD40010000        <1>       mov    ebp, 320 ; 320*200, 320*240
3775 0000E58B 20E4              <1>       and    ah, ah
3776                            <1>       ;jz    short sysvideo_13_0
3777 0000E58D 740E              <1>       jz     short sysvideo_14 ; 28/12/2020
3778 0000E58F D1E5              <1>       shl    ebp, 1 ; 640*200, 640*400, 640*480
3779 0000E591 80FC02            <1>       cmp    ah, 2
3780                            <1>       ;jb    short sysvideo_13_0
3781 0000E594 7207              <1>       jb     short sysvideo_14 ; 28/12/2020
3782                            <1>       ;;ja   sysret ; invalid
3783 0000E596 77DA              <1>       ja     short sysvideo_12_0 ; 26/12/2020
3784                            <1>       ; 800*600
3785 0000E598 6681C5A000        <1>       add    bp, 160 ; 800
3786                            <1>
3787                            <1> ;sysvideo_13_0:
3788                            <1> sysvideo_14:
3789                            <1>       ; 28/12/2020
3790                            <1>       ; ebp = screen width (320, 640, 800)
3791 0000E59D B800000A00        <1>       mov    eax, 0A0000h ; Video memory
3792 0000E5A2 50                <1>       push   eax ; * ; save linear frame buffer address
3793 0000E5A3 B800000100        <1>       mov    eax, 65536   ; video memory size
3794 0000E5A8 B708              <1>       mov    bh, 8  ; 02/01/2020
3795 0000E5AA E992000000        <1>       jmp    sysvideo_15_6_0
3796                            <1>
3797                            <1> sysvideo_15:
3798                            <1>       ; 28/12/2020
3799                            <1>
3800 0000E5AF 80FF02            <1>       cmp    bh, 2
3801 0000E5B2 0F87B8160000      <1>         ja      sysvideo_16
3802                            <1>
3803                            <1>       ; 01/01/2021
3804                            <1>       ; 27/12/2020
3805                            <1>       ; 26/12/2020
3806                            <1>       ; 25/12/2020 (TRDOS 386 v2.0.3)
3807                            <1>       ;
3808                            <1>       ; BH = 2 = SVGA (VESA VBE) Graphics mode (LFB) data transfers
3809                            <1>
3810                            <1>       ; 25/12/2020
3811                            <1>       ; resolution table entry will be saved into EBP register
3812                            <1>
3813                            <1>       ;cmp   bl, 13
3814                            <1>       ; 01/01/2021
3815 0000E5B8 80FB0F            <1>       cmp    bl, 15 ; cmp bl,0Fh
3816 0000E5BB 77B5              <1>       ja     short sysvideo_12_0 ; invalid (unknown) sub function
3817                            <1>
3818 0000E5BD 803D[54090000]02  <1>       cmp    byte [vbe3], 2 ; VESA VBE 3 video bios
3819                            <1>                           ; or BOCHS/QEMU/VIRTUALBOX emu video bios
3820 0000E5C4 7243              <1>       jb     short sysvideo_15_4 ; no, nothing to do !
3821 0000E5C6 770B              <1>       ja     short sysvideo_15_0 ; yes
3822                            <1>
3823                            <1>       ; Only Bochs/Plex86 (emu) vbe2 video bios is usable in pmid
3824                            <1>       ; (if [vbe3] = 2)
3825 0000E5C8 A0[55090000]      <1>       mov    al, [vbe2bios] ; Bochs vbios sign is from C0h to C5h
3826 0000E5CD 24F0              <1>       and    al, 0F0h
3827 0000E5CF 3CC0              <1>       cmp    al, 0C0h
3828 0000E5D1 7536              <1>       jne    short sysvideo_15_4 ; unknown (vbe2) video bios
3829                            <1> sysvideo_15_0:
3830                            <1>       ; 01/01/2021
3831 0000E5D3 80FB0E            <1>       cmp    bl, 14
3832 0000E5D6 7307              <1>       jnb    short sysvideo_15_0_0
3833                            <1>       ;
3834 0000E5D8 89D8              <1>       mov    eax, ebx ; hw of ebx is vesa vbe video mode
3835 0000E5DA C1E810            <1>       shr    eax, 16 ; ax = vesa vbe video mode
3836 0000E5DD 7513              <1>       jnz    short sysvideo_15_2
3837                            <1>       ; ax = 0
3838                            <1> sysvideo_15_0_0:
3839                            <1>       ; check & use current video mode
3840 0000E5DF 803D[BA6F0000]FF  <1>       cmp    byte [CRT_MODE], 0FFh ; extended (SVGA) mode ?
3841 0000E5E6 7521              <1>       jne    short sysvideo_15_4 ; no
3842                            <1> sysvideo_15_1:
3843                            <1>       ; use current vbe (svga) video mode
3844 0000E5E8 66A1[1E120300]    <1>       mov    ax, [video_mode] ; extended (SVGA, VESA VBE) mode
3845 0000E5EE 6625FF01          <1>       and    ax, 1FFh ; vesa vbe video mode: 1XXh
3846                            <1> sysvideo_15_2:
3847 0000E5F2 56                <1>       push   esi
3848 0000E5F3 BE[46730000]      <1>       mov    esi, b_vbe_modes ; vbe mode table (in 'vidata.s')
```

```
                            <1> sysvideo_15_3:
0000E5F8 663B06             <1>       cmp    ax, [esi]
0000E5FB 7411               <1>       je     short sysvideo_15_5
0000E5FD 83C608             <1>       add    esi, 8 ; vbe mode table entry size
0000E600 81FE[06740000]     <1>       cmp    esi, end_of_b_vbe_modes
0000E606 72F0               <1>       jb     short sysvideo_15_3
0000E608 5E                 <1>       pop    esi
                            <1> sysvideo_15_4:
                            <1>       ; desired video mode is not a valid (implemented)
                            <1>       ;       extended (VESA VBE, SVGA) video mode
                            <1>       ;
                            <1>       ; nothing to do !
                            <1>
                            <1>       ; [u.r0] = 0  ; return value of EAX
0000E609 E9DEF2FFFF         <1>       jmp    sysret
                            <1>
                            <1> sysvideo_15_5:
0000E60E 89F5               <1>       mov    ebp, esi
                            <1>
0000E610 5E                 <1>       pop    esi
                            <1>
                            <1>       ; get LFB address
0000E611 A1[2C120300]       <1>       mov    eax, [LFB_ADDR] ; [LFB_Info+LFBINFO.LFB_addr]
0000E616 09C0               <1>       or     eax, eax
0000E618 7509               <1>       jnz    short sysvideo_15_6
0000E61A 66A1[E30E0000]     <1>       mov    ax, [def_LFB_addr] ; default LFB addr
                            <1>                                 ; (for vbe mode 118h)
0000E620 C1E010             <1>       shl    eax, 16
                            <1>       ; 27/12/2020
                            <1>       ;jz    short sysvideo_15_4
                            <1> sysvideo_15_6:
0000E623 50                 <1>       push   eax ; * ; save linear frame buffer address
                            <1>
                            <1>       ; 27/12/2020
                            <1>       ; 26/12/2020
0000E624 8B4502             <1>       mov    eax, [ebp+2] ; width, height
                            <1>       ; 28/12/2020
0000E627 8A7D06             <1>       mov    bh, byte [ebp+6] ; bpp
0000E62A 0FB7E8             <1>       movzx  ebp, ax      ; width
                            <1>
0000E62D 52                 <1>       push   edx ; ***
0000E62E 0FB7D0             <1>       movzx  edx, ax  ; width
0000E631 C1E810             <1>       shr    eax, 16  ; height
0000E634 F7E2               <1>       mul    edx
                            <1>       ; eax = linear frame buffer size (pixels)
0000E636 5A                 <1>       pop    edx ; ***
                            <1>       ;mov   bh, byte [ebp+6] ; bpp
                            <1>
0000E637 80FF08             <1>       cmp    bh, 8
0000E63A 7405               <1>       je     short sysvideo_15_6_0
                            <1>
0000E63C E935020000         <1>       jmp    sysvideo_15_39 ; 16, 24, 32 bpp
                            <1>
                            <1> sysvideo_15_6_0:
                            <1>       ; eax = linear frame buffer (VIDEO PAGE) size
                            <1>       ; bh = bits per pixel (bpp) = 8
                            <1>
0000E641 08DB               <1>       or     bl, bl
0000E643 750E               <1>       jnz    short sysvideo_15_7
                            <1>
0000E645 5F                 <1>       pop    edi ; * ; LFB address
                            <1>
0000E646 A3[64030300]       <1>       mov    [u.r0], eax
                            <1>
                            <1>       ; BL = 0 = Fill color (color in CL)
0000E64B 91                 <1>       xchg   eax, ecx ; al = color, ecx = byte count
                            <1>
                            <1>       ; 8 bit pixels
0000E64C F3AA               <1>       rep    stosb
                            <1> sysvideo_15_6_1:
                            <1>       ; [u.r0] = transfer (fill) count in bytes
0000E64E E999F2FFFF         <1>       jmp    sysret
                            <1>
                            <1> sysvideo_15_7:
                            <1>       ; eax = linear frame buffer (VIDEO PAGE) size
                            <1>
0000E653 80FB02             <1>       cmp    bl, 2
0000E656 7725               <1>       ja     short sysvideo_15_10
0000E658 7217               <1>       jb     short sysvideo_15_9
                            <1>
                            <1>       ; BL = 2 = system to user video/display page transfer
0000E65A 5E                 <1>       pop    esi ; * ; LFB address
0000E65B 89CF               <1>       mov    edi, ecx ; user's buffer
                            <1> sysvideo_15_7_0:
0000E65D 89C1               <1>       mov    ecx, eax ; LFB video page size (< LFB size)
0000E65F E8602C0000         <1>       call   transfer_to_user_buffer ; fast transfer
                            <1> sysvideo_15_8:
                            <1>       ;jc    sysret ; [u.r0] = 0
0000E664 72E8               <1>       jc     short sysvideo_15_6_1
0000E666 890D[64030300]     <1>       mov    [u.r0], ecx ; actual transfer count
0000E66C E97BF2FFFF         <1>       jmp    sysret
                            <1>
                            <1> sysvideo_15_9:
                            <1>       ; BL = 1 = user to system video/display page transfer
0000E671 5F                 <1>       pop    edi ; * ; LFB address
0000E672 89CE               <1>       mov    esi, ecx ; user's buffer
                            <1> sysvideo_15_9_0:
0000E674 89C1               <1>       mov    ecx, eax ; LFB video page size (< LFB size)
0000E676 E8932C0000         <1>       call   transfer_from_user_buffer ; fast transfer
0000E67B EBE7               <1>       jmp    short sysvideo_15_8
                            <1>
                            <1> sysvideo_15_10:
0000E67D 80FB03             <1>       cmp    bl, 3
0000E680 7762               <1>       ja     short sysvideo_15_16
                            <1>
```

```
                                <1>       ; BL = 3 = NOT bits in window (ECX, EDX)
0000E682 5E                     <1>       pop    esi ; * ; LFB address
                                <1>
0000E683 39CA                   <1>       cmp    edx, ecx ; bottom-right > top-left ?
0000E685 7318                   <1>       jnb    short sysvideo_15_12 ; window
                                <1>
0000E687 09D2                   <1>       or     edx, edx
0000E689 7554                   <1>       jnz    short sysvideo_15_15  ; invalid !
                                <1>
                                <1>       ; full screen (update) ; edx = 0, ecx > 0
0000E68B 89C1                   <1>       mov    ecx, eax ; LFB page size (byte count)
0000E68D C1E902                 <1>       shr    ecx, 2  ; dword 'NOT'
0000E690 A3[64030300]           <1>       mov    [u.r0], eax
                                <1> sysvideo_15_11:
0000E695 F716                   <1>       not    dword [esi]
0000E697 AD                     <1>       lodsd  ; add esi, 4
0000E698 E2FB                   <1>       loop   sysvideo_15_11
0000E69A E94DF2FFFF             <1>       jmp    sysret
                                <1>
                                <1> sysvideo_15_12:
                                <1>       ; 29/12/2020
                                <1>       ; fix dx to bp-1 if dx >= bp
0000E69F E885140000             <1>       call   sysvideo_15_200
                                <1>       ;
                                <1>       ; 29/12/2020
0000E6A4 89F7                   <1>       mov    edi, esi
0000E6A6 01C7                   <1>       add    edi, eax ; end byte of video page + 1
0000E6A8 0FB7C2                 <1>       movzx  eax, dx ; bottom right column
0000E6AB 6629C8                 <1>       sub    ax, cx  ; - top left column
0000E6AE 722F                   <1>         jb      short sysvideo_15_15 ; invalid
0000E6B0 40                     <1>       inc    eax ; same column no == 1 column
0000E6B1 50                     <1>       push   eax ; byte count per window row
0000E6B2 52                     <1>       push   edx
                                <1>       ;;movzx      ebx, word [ebp+2] ; screen width
                                <1>       ;; 28/12/2020
                                <1>       ;mov   ebx, ebp ; screen width
                                <1>       ;mov   eax, ecx
                                <1>       ;shr   eax, 16      ; top row
                                <1>       ;mul   ebx
                                <1>       ;mov   dx, cx ; top left column
                                <1>       ;add   eax, edx
                                <1>       ; 30/12/2020
                                <1>       ; calculate start offset
0000E6B3 E87C140000             <1>       call   sysvideo_15_202
0000E6B8 01C6                   <1>       add    esi, eax ; start address
0000E6BA 59                     <1>       pop    ecx ; edx
                                <1>       ;mov   eax, ecx
                                <1>       ;shr   eax, 16 ; bottom row  ; zero based
                                <1>       ;mul   ebx
                                <1>       ;mov   dx, cx ; bottom right column ; 0 based
                                <1>       ;add   eax, edx
                                <1>       ; 30/12/2020
                                <1>       ; calculate start offset
0000E6BB E876140000             <1>       call   sysvideo_15_203
0000E6C0 5A                     <1>       pop    edx ; byte count per window row
                                <1>       ; eax = the last byte to be converted
0000E6C1 01F0                   <1>       add    eax, esi
0000E6C3 39C7                   <1>       cmp    edi, eax
0000E6C5 7618                   <1>       jna    short sysvideo_15_15 ; out of video page
0000E6C7 89C7                   <1>       mov    edi, eax ; stop address
0000E6C9 29D3                   <1>       sub    ebx, edx ; screen width - window width
0000E6CB 4E                     <1>       dec    esi
                                <1> sysvideo_15_13:
0000E6CC 89D1                   <1>       mov    ecx, edx ; window width (pixel count)
                                <1> sysvideo_15_14:
0000E6CE 46                     <1>       inc    esi
0000E6CF F616                   <1>       not    byte [esi]
0000E6D1 E2FB                   <1>       loop   sysvideo_15_14
0000E6D3 0115[64030300]         <1>       add    [u.r0], edx
0000E6D9 01DE                   <1>       add    esi, ebx ; add difference for next row
                                <1>       ;
0000E6DB 39FE                   <1>       cmp    esi, edi ; stop address (included in loop)
0000E6DD 76ED                   <1>       jna    short sysvideo_15_13
                                <1>       ;jb    short sysvideo_15_13
                                <1> sysvideo_15_15:
0000E6DF E908F2FFFF             <1>       jmp    sysret
                                <1>
                                <1> sysvideo_15_16:
0000E6E4 80FB04                 <1>       cmp    bl, 4
0000E6E7 7771                   <1>         ja   short sysvideo_15_20
                                <1>
                                <1>       ; BL = 4 = window copy (system to system)
                                <1>
                                <1>       ; eax = LFB page size in bytes
                                <1>
0000E6E9 5B                     <1>       pop    ebx ; * ; LFB address
                                <1>
0000E6EA 39DE                   <1>       cmp    esi, ebx
0000E6EC 72F1                   <1>       jb     short sysvideo_15_15
                                <1>
0000E6EE 39DF                   <1>       cmp    edi, ebx
0000E6F0 72ED                   <1>       jb     short sysvideo_15_15
0000E6F2 01C3                   <1>       add    ebx, eax ; **
0000E6F4 39DE                   <1>       cmp    esi, ebx
0000E6F6 73E7                   <1>       jnb    short sysvideo_15_15
0000E6F8 39DF                   <1>       cmp    edi, ebx
0000E6FA 73E3                   <1>       jnb    short sysvideo_15_15
                                <1>
0000E6FC 39CA                   <1>       cmp    edx, ecx ; bottom-right > top-left ?
0000E6FE 7312                   <1>       jnb    short sysvideo_15_17 ; window
                                <1>
0000E700 09D2                   <1>       or     edx, edx
0000E702 75DB                   <1>       jnz    short sysvideo_15_15 ; invalid !
                                <1>
0000E704 A3[64030300]           <1>       mov    [u.r0], eax
```

```
4059 0000E709 89C1              <1>       mov    ecx, eax ; pixel (byte) count
4060                            <1>       ; full screen copy ; edx = 0, ecx > 0
4061 0000E70B F3A4              <1>       rep    movsb
4062 0000E70D E9DAF1FFFF        <1>       jmp    sysret
4063                            <1> sysvideo_15_17:
4064                            <1>       ; 29/12/2020
4065                            <1>       ; fix dx to bp-1 if dx >= bp
4066 0000E712 E812140000        <1>       call   sysvideo_15_200
4067                            <1>       ;
4068                            <1>       ; 29/12/2020
4069 0000E717 6639CA            <1>       cmp    dx, cx
4070 0000E71A 72C3              <1>       jb     short sysvideo_15_15 ; invalid
4071                            <1>
4072 0000E71C 53                <1>       push   ebx ; the last byte of LFB + 1
4073 0000E71D 29C3              <1>       sub    ebx, eax ; **
4074 0000E71F 53                <1>       push   ebx ; LFB start/base address
4075                            <1>
4076 0000E720 0FB7C2            <1>       movzx  eax, dx ; bottom right column
4077 0000E723 6629C8            <1>       sub    ax, cx  ; - top left column
4078 0000E726 40                <1>       inc    eax ; same column no == 1 column
4079 0000E727 50                <1>       push   eax ; byte (pixel) count per window row
4080 0000E728 52                <1>       push   edx
4081                            <1>       ;;movzx      ebx, word [ebp+2] ; screen width
4082                            <1>       ;; 28/12/2020
4083                            <1>       ;mov   ebx, ebp ; screen width
4084                            <1>       ;mov   eax, ecx
4085                            <1>       ;shr   eax, 16      ; top row
4086                            <1>       ;mul   ebx
4087                            <1>       ;mov   dx, cx ; top left column
4088                            <1>       ;add   eax, edx
4089                            <1>       ; 30/12/2020
4090                            <1>       ; calculate start offset
4091 0000E729 E806140000        <1>       call   sysvideo_15_202
4092 0000E72E 01C7              <1>       add    edi, eax ; start address
4093 0000E730 01C6              <1>       add    esi, eax
4094 0000E732 59                <1>       pop    ecx ; edx
4095                            <1>       ;mov   eax, ecx
4096                            <1>       ;shr   eax, 16 ; bottom row  ; zero based
4097                            <1>       ;mul   ebx
4098                            <1>       ;mov   dx, cx ; bottom right column ; 0 based
4099                            <1>       ;add   eax, edx
4100                            <1>       ; 30/12/2020
4101                            <1>       ; calculate start offset
4102 0000E733 E8FE130000        <1>       call   sysvideo_15_203
4103 0000E738 5A                <1>       pop    edx ; byte (pixel) count per window row
4104 0000E739 59                <1>       pop    ecx ; LFB start/base address
4105 0000E73A 01C8              <1>       add    eax, ecx ; stop address
4106 0000E73C 59                <1>       pop    ecx ; the last byte of LFB + 1
4107 0000E73D 39C8              <1>       cmp    eax, ecx
4108 0000E73F 739E              <1>       jnb    short sysvideo_15_15
4109                            <1>
4110                            <1>       ;push  edi ; start address
4111                            <1>       ;push  eax ; stop address (included)
4112 0000E741 29D3              <1>       sub    ebx, edx ; screen width - window width
4113                            <1> sysvideo_15_18:
4114 0000E743 89D1              <1>       mov    ecx, edx
4115 0000E745 F3A4              <1>       rep    movsb
4116 0000E747 0115[64030300]    <1>       add    [u.r0], edx ; window width (byte count)
4117                            <1>       ;or    ebx, ebx
4118                            <1>       ;jz    short sysvideo_15_19
4119 0000E74D 01DF              <1>       add    edi, ebx ; add difference (for next row)
4120 0000E74F 01DE              <1>       add    esi, ebx
4121                            <1> ;sysvideo_15_19:
4122                            <1>       ;cmp   edi, [esp] ; stop addr (included in loop)
4123 0000E751 39C7              <1>       cmp    edi, eax
4124 0000E753 76EE              <1>       jna    short sysvideo_15_18
4125                            <1> sysvideo_15_19:
4126                            <1>       ;pop   ebx ; stop address
4127                            <1>       ;pop   edi ; start address
4128 0000E755 E992F1FFFF        <1>       jmp    sysret
4129                            <1>
4130                            <1> sysvideo_15_20:
4131 0000E75A 80FB05            <1>       cmp    bl, 5
4132 0000E75D 7752              <1>         ja      short sysvideo_15_25
4133                            <1>
4134                            <1>       ; BL = 5 = window copy (user to system)
4135                            <1>
4136                            <1>       ; eax = LFB page size in bytes
4137                            <1>
4138 0000E75F 5F                <1>       pop    edi ; * ; LFB address
4139                            <1>
4140                            <1>       ; esi = user buffer (in user's memory space)
4141 0000E760 39CA              <1>       cmp    edx, ecx ; bottom-right > top-left ?
4142 0000E762 7309              <1>              jnb    short sysvideo_15_21 ; window
4143                            <1>
4144 0000E764 09D2              <1>       or     edx, edx
4145                            <1>       ;jnz   sysret  ; invalid !
4146 0000E766 75ED              <1>       jnz    short sysvideo_15_19
4147                            <1>
4148                            <1>       ; full screen copy ; edx = 0, ecx > 0
4149 0000E768 E907FFFFFF        <1>       jmp    sysvideo_15_9_0 ; full screen copy
4150                            <1>
4151                            <1> sysvideo_15_21:
4152                            <1>       ; 29/12/2020
4153                            <1>       ; fix dx to bp-1 if dx >= bp
4154 0000E76D E8B7130000        <1>       call   sysvideo_15_200
4155                            <1>       ;
4156                            <1>       ; 29/12/2020
4157 0000E772 6639CA            <1>       cmp    dx, cx ; righ pos >= left pos
4158 0000E775 72DE              <1>       jb     short sysvideo_15_19 ; wrong pos input
4159                            <1>
4160                            <1>       ; 01/01/2021
4161 0000E777 01F8              <1>       add    eax, edi
4162 0000E779 50                <1>       push   eax ; the last byte of LFB + 1
4163 0000E77A 57                <1>       push   edi ; LFB start/base address
```

```
                                <1>
0000E77B 0FB7C2                 <1>      movzx  eax, dx ; bottom right column
0000E77E 6629C8                 <1>      sub    ax, cx  ; - top left column
0000E781 40                     <1>      inc    eax ; same column no == 1 column
0000E782 50                     <1>      push   eax ; byte count per window row
                                <1>
0000E783 52                     <1>      push   edx
                                <1>      ;movzx ebx, word [ebp+2] ; screen width
                                <1>      ; 30/12/2020
                                <1>      ; calculate start offset
                                <1>      ;mov   ebx, ebp ; screen width
                                <1>      ;mov   eax, ecx
                                <1>      ;shr   eax, 16  ; top row
                                <1>      ;mul   ebx
                                <1>      ;mov   dx, cx ; top left column
                                <1>      ;add   eax, edx
0000E784 E8AB130000             <1>      call   sysvideo_15_202
0000E789 01C7                   <1>      add    edi, eax ; start address
0000E78B 59                     <1>      pop    ecx ; edx
                                <1>      ;mov   eax, ecx
                                <1>      ;shr   eax, 16 ; bottom row
                                <1>      ;mul   ebx
                                <1>      ;mov   dx, cx ; bottom right column
                                <1>      ;add   eax, edx
                                <1>      ; 30/12/2020
                                <1>      ; calculate end offset
0000E78C E8A5130000             <1>      call   sysvideo_15_203
0000E791 5A                     <1>      pop    edx ; byte count per window row
0000E792 59                     <1>      pop    ecx ; LFB start/base address
0000E793 01C8                   <1>      add    eax, ecx ; stop address
0000E795 59                     <1>      pop    ecx ; the last byte of LFB + 1
                                <1>      ; 29/12/2020
0000E796 39C8                   <1>      cmp    eax, ecx ; row >= vertical resolution ?
0000E798 7312                   <1>      jnb    short sysvideo_15_24 ; yes !, wrong !
                                <1>
                                <1>      ;push  edi ; start address
0000E79A 50                     <1>      push   eax ; stop address (included)
                                <1> sysvideo_15_22:
0000E79B 89D1                   <1>      mov    ecx, edx ; byte count
                                <1>      ; user to system video/display page window transfer
                                <1>      ; esi =      user buffer
0000E79D E86C2B0000             <1>      call   transfer_from_user_buffer ; fast transfer
0000E7A2 7207                   <1>      jc     short sysvideo_15_23
                                <1>             ; ecx = actual transfer count (= ecx input)
                                <1>      ;add   [u.r0], ecx
                                <1>      ;add   edi, ebx ; next row in video memory window
                                <1>      ;add   esi, ecx ; next window row in user's buffer
                                <1>      ;cmp   edi, [esp] ; stop addr (included in loop)
                                <1>      ; 30/12/2020
0000E7A4 E89A130000             <1>      call   sysvideo_15_204
0000E7A9 76F0                   <1>      jna    short sysvideo_15_22
                                <1> sysvideo_15_23:
0000E7AB 5B                     <1>      pop    ebx ; stop address
                                <1>      ;pop   edi ; start address
                                <1> sysvideo_15_24:
0000E7AC E93BF1FFFF             <1>      jmp    sysret
                                <1>
                                <1> sysvideo_15_25:
0000E7B1 80FB06                 <1>      cmp    bl, 6
0000E7B4 7752                   <1>      ja     short sysvideo_15_28
                                <1>
                                <1>      ; BL = 6 = window copy (system to user)
                                <1>
                                <1>      ; eax = LFB page size in bytes
                                <1>
0000E7B6 89F7                   <1>      mov    edi, esi ; user buffer
                                <1>
0000E7B8 5E                     <1>      pop    esi ; * ; LFB address
                                <1>
                                <1>      ; edi = user buffer (in user's memory space)
0000E7B9 39CA                   <1>      cmp    edx, ecx ; bottom-right > top-left ?
0000E7BB 7309                   <1>        jnb     short sysvideo_15_26
                                <1>
0000E7BD 21D2                   <1>      and    edx, edx
                                <1>      ;jnz   sysret ; invalid !
0000E7BF 75EB                   <1>      jnz    short sysvideo_15_24
                                <1>
                                <1>      ; full screen copy  ; edx = 0, ecx > 0
0000E7C1 E997FEFFFF             <1>      jmp    sysvideo_15_7_0
                                <1>
                                <1> sysvideo_15_26:
                                <1>      ; 29/12/2020
                                <1>      ; fix dx to bp-1 if dx >= bp
0000E7C6 E85E130000             <1>      call   sysvideo_15_200
                                <1>      ;
                                <1>      ; 29/12/2020
0000E7CB 6639CA                 <1>      cmp    dx, cx ; righ pos >= left pos
0000E7CE 72DC                   <1>      jb     short sysvideo_15_24 ; wrong pos input
                                <1>
0000E7D0 89F3                   <1>      mov    ebx, esi ; LFB address
0000E7D2 01D8                   <1>      add    eax, ebx
0000E7D4 50                     <1>      push   eax ; the last byte of LFB + 1
0000E7D5 53                     <1>      push   ebx ; LFB start/base address
                                <1>
0000E7D6 0FB7C2                 <1>      movzx  eax, dx ; bottom right column
0000E7D9 6629C8                 <1>      sub    ax, cx  ; - top left column
0000E7DC 40                     <1>             inc    eax ; same column no == 1 column
0000E7DD 50                     <1>      push   eax ; byte count per window row
                                <1>
0000E7DE 52                     <1>      push   edx
                                <1>      ;movzx ebx, word [ebp+2] ; screen width
                                <1>      ; 30/12/2020
                                <1>      ; calculate start offset
                                <1>      ;mov   ebx, ebp ; screen width
                                <1>      ;mov   eax, ecx
```

```
                                <1>       ;shr   eax, 16  ; top row
                                <1>       ;mul   ebx
                                <1>       ;mov   dx, cx ; top left column
                                <1>       ;add   eax, edx
0000E7DF E850130000             <1>       call   sysvideo_15_202
0000E7E4 01C6                   <1>       add    esi, eax ; start address
0000E7E6 59                     <1>       pop    ecx ; edx
                                <1>       ;mov   eax, ecx
                                <1>       ;shr   eax, 16 ; bottom row
                                <1>       ;mul   ebx
                                <1>       ;mov   dx, cx ; bottom right column
                                <1>       ;add   eax, edx
                                <1>       ; 30/12/2020
                                <1>       ; calculate end offset
0000E7E7 E84A130000             <1>       call   sysvideo_15_203
0000E7EC 5A                     <1>       pop    edx ; byte count per window row
0000E7ED 59                     <1>       pop    ecx ; LFB start/base address
0000E7EE 01C8                   <1>       add    eax, ecx ; stop address
0000E7F0 59                     <1>       pop    ecx ; the last byte of LFB + 1
                                <1>       ; 29/12/2020
0000E7F1 39C8                   <1>       cmp    eax, ecx ; row >= vertical resolution ?
0000E7F3 73B7                   <1>       jnb    short sysvideo_15_24 ; yes !, wrong !
                                <1>
                                <1>       ;push  esi ; start address
0000E7F5 50                     <1>       push   eax ; stop address (included)
                                <1> sysvideo_15_27:
0000E7F6 89D1                   <1>       mov    ecx, edx ; byte count
                                <1>       ; user to system video/display page window transfer
                                <1>       ; esi =      user buffer
0000E7F8 E8C72A0000             <1>       call   transfer_to_user_buffer ; fast transfer
                                <1>       ;jc    short sysvideo_15_27_0
0000E7FD 72AC                   <1>       jc     short sysvideo_15_23
                                <1>       ;add   [u.r0], ecx
                                <1>       ;add   edi, ebx ; next row in video memory window
                                <1>       ;add   esi, ecx ; next window row in user's buffer
                                <1>       ;cmp   edi, [esp] ; stop addr (included in loop)
                                <1>       ; 30/12/2020
0000E7FF E83F130000             <1>       call   sysvideo_15_204
0000E804 76F0                   <1>       jna    short sysvideo_15_27
0000E806 EBA3                   <1>       jmp    short sysvideo_15_23
                                <1>
                                <1> sysvideo_15_28:
                                <1>       ; bl = 7, 8, 9, 10, 11, 12, 13
                                <1>       ; (bl > 13 is invalid for current kernel version)
                                <1>
                                <1>       ;cmp   bl, 13
                                <1>       ;ja    short sysvideo_15_23
                                <1>
                                <1>       ; 01/01/2021
0000E808 80FB0D                 <1>       cmp    bl, 13
0000E80B 0F8722080000           <1>       ja     sysvideo_15_138
                                <1>
                                <1>       ; eax = LFB video page size (w*h) in bytes
                                <1>
0000E811 5E                     <1>       pop    esi ; * ; LFB base/start address
0000E812 91                     <1>       xchg   eax, ecx ; ecx = byte (pixel) count
                                <1>                       ; al = byte for and/or/xor
0000E813 890D[64030300]         <1>       mov    [u.r0], ecx ; return with
                                <1>                          ; byte count of display page
                                <1>                          ; (eax > 0 -> OK)
0000E819 80FB0A                 <1>       cmp    bl, 10
0000E81C 720C                   <1>       jb     short sysvideo_15_29 ; 7, 8, 9
0000E81E 7414                   <1>       je     short sysvideo_15_31 ; 10
0000E820 80FB0C                 <1>       cmp    bl, 12
0000E823 7216                   <1>       jb     short sysvideo_15_32 ; 11
0000E825 741B                   <1>       je     short sysvideo_15_33 ; 12
                                <1>
                                <1>       ; BL = 13 = NOT display page bytes
0000E827 C1E902                 <1>       shr    ecx, 2 ; dword count
                                <1> sysvideo_15_29:
0000E82A F716                   <1>       not    dword [esi]
0000E82C AD                     <1>       lodsd
0000E82D E2FB                   <1>       loop   sysvideo_15_29
                                <1>
                                <1> sysvideo_15_30:
0000E82F E9B8F0FFFF             <1>       jmp    sysret
                                <1>
                                <1>       ; BL = 10 = INC display page bytes
                                <1> sysvideo_15_31:
0000E834 FE06                   <1>       inc    byte [esi]
0000E836 46                     <1>       inc    esi
0000E837 E2FB                   <1>       loop   sysvideo_15_31
0000E839 EBF4                   <1>       jmp    short sysvideo_15_30
                                <1>
                                <1>       ; BL = 11 = DEC display page bytes
                                <1> sysvideo_15_32:
0000E83B FE0E                   <1>       dec    byte [esi]
0000E83D 46                     <1>       inc    esi
0000E83E E2FB                   <1>       loop   sysvideo_15_32
0000E840 EBED                   <1>       jmp    short sysvideo_15_30
                                <1>
                                <1>       ; BL = 11 = NEG display page bytes
                                <1> sysvideo_15_33:
0000E842 F61E                   <1>       neg    byte [esi]
0000E844 46                     <1>       inc    esi
0000E845 E2FB                   <1>       loop   sysvideo_15_33
0000E847 EBE6                   <1>       jmp    short sysvideo_15_30
                                <1>
                                <1> sysvideo_15_34:
                                <1>       ; bl = 7, 8, 9
                                <1>       ; ecx = byte (pixel) count
0000E849 C1E902                 <1>       shr    ecx, 2 ; dword count
0000E84C 88C4                   <1>       mov    ah, al
0000E84E 6689C2                 <1>       mov    dx, ax
0000E851 C1E210                 <1>       shl    edx, 16
```

```
4374 0000E854 6689C2              <1>       mov    dx, ax
4375                              <1>              ; edx = 4 bytes for and/or/xor
4376                              <1>
4377 0000E857 80FB08              <1>       cmp    bl, 8
4378 0000E85A 7409                <1>       je     short sysvideo_15_36
4379 0000E85C 770E                <1>       ja     short sysvideo_15_37
4380                              <1>
4381                              <1>       ; BL = 7 = AND display page bytes with CL
4382                              <1> sysvideo_15_35:
4383 0000E85E 2116                <1>       and    [esi], edx
4384 0000E860 AD                  <1>       lodsd
4385 0000E861 E2FB                <1>       loop   sysvideo_15_35
4386                              <1>       ;jmp   sysret
4387 0000E863 EB0C                <1>       jmp    short sysvideo_15_38
4388                              <1>
4389                              <1>       ; BL = 8 = OR display page bytes with CL
4390                              <1> sysvideo_15_36:
4391 0000E865 0916                <1>       or     [esi], edx
4392 0000E867 AD                  <1>       lodsd
4393 0000E868 E2FB                <1>       loop   sysvideo_15_36
4394                              <1>       ;jmp   sysret
4395 0000E86A EB05                <1>       jmp    short sysvideo_15_38
4396                              <1>
4397                              <1>       ; BL = 9 = XOR display page bytes with CL
4398                              <1> sysvideo_15_37:
4399 0000E86C 3116                <1>       xor    [esi], edx
4400 0000E86E AD                  <1>       lodsd
4401 0000E86F E2FB                <1>       loop   sysvideo_15_37
4402                              <1> sysvideo_15_38:
4403 0000E871 E976F0FFFF          <1>       jmp    sysret
4404                              <1>
4405                              <1> sysvideo_15_39:
4406                              <1>       ; bh > 8 (bpp, 16 or 24 or 32)
4407                              <1>
4408 0000E876 80FF18              <1>       cmp    bh, 24
4409 0000E879 0F8760020000        <1>       ja     sysvideo_15_72
4410 0000E87F 0F84C6040000        <1>       je     sysvideo_15_105
4411                              <1>
4412                              <1>       ; bh = bits per pixel (bpp) = 16
4413                              <1>
4414                              <1>       ; eax = linear frame buffer (VIDEO PAGE) pixel count
4415                              <1>       ;      (half of video page size in bytes)
4416                              <1>
4417 0000E885 D1E0                <1>       shl    eax, 1 ; byte count = 2 * pixel count
4418                              <1>
4419                              <1>       ; sub function for sysvideo function bh = 2
4420 0000E887 20DB                <1>       and    bl, bl ; 0 ?
4421 0000E889 7511                <1>       jnz    short sysvideo_15_40 ; no
4422                              <1>
4423                              <1>       ; BL = 0 = Fill color (color in CX)
4424                              <1>
4425 0000E88B 5F                  <1>       pop    edi ; * ; LFB address
4426                              <1>
4427 0000E88C A3[64030300]        <1>       mov    [u.r0], eax ; return value to user
4428                              <1>
4429 0000E891 91                  <1>       xchg   eax, ecx ; ax = color, ecx = byte count
4430                              <1>
4431                              <1>       ; 16 bit pixels
4432 0000E892 D1E9                <1>       shr    ecx, 1
4433                              <1>       ; ecx = pixel count
4434                              <1>
4435 0000E894 F366AB              <1>       rep    stosw
4436                              <1>
4437                              <1>       ; [u.r0] = transfer (fill) count in bytes
4438 0000E897 E950F0FFFF          <1>       jmp    sysret
4439                              <1>
4440                              <1> sysvideo_15_40:
4441                              <1>       ; eax = linear frame buffer (VIDEO PAGE) byte count
4442                              <1>       ;       (2 * pixel count)
4443                              <1>
4444 0000E89C 80FB02              <1>       cmp    bl, 2
4445 0000E89F 7725                <1>       ja     short sysvideo_15_45
4446 0000E8A1 7217                <1>       jb     short sysvideo_15_43
4447                              <1>
4448                              <1>       ; BL = 2 = system to user video/display page transfer
4449 0000E8A3 5E                  <1>       pop    esi ; * ; LFB address
4450 0000E8A4 89CF                <1>       mov    edi, ecx ; user's buffer
4451                              <1> sysvideo_15_41:
4452 0000E8A6 89C1                <1>       mov    ecx, eax ; LFB video page byte count
4453 0000E8A8 E8172A0000          <1>       call   transfer_to_user_buffer ; fast transfer
4454                              <1> sysvideo_15_42:
4455                              <1>       ;jc    sysret ; [u.r0] = 0
4456 0000E8AD 72C2                <1>       jc     short sysvideo_15_38
4457 0000E8AF 890D[64030300]      <1>       mov    [u.r0], ecx ; actual transfer count
4458 0000E8B5 E932F0FFFF          <1>       jmp    sysret
4459                              <1>
4460                              <1> sysvideo_15_43:
4461                              <1>       ; BL = 1 = user to system video/display page transfer
4462 0000E8BA 5F                  <1>       pop    edi ; * ; LFB address
4463 0000E8BB 89CE                <1>       mov    esi, ecx ; user's buffer
4464                              <1> sysvideo_15_44:
4465 0000E8BD 89C1                <1>       mov    ecx, eax ; LFB video page byte count
4466 0000E8BF E84A2A0000          <1>       call   transfer_from_user_buffer ; fast transfer
4467 0000E8C4 EBE7                <1>       jmp    short sysvideo_15_42
4468                              <1>
4469                              <1> sysvideo_15_45:
4470 0000E8C6 80FB03              <1>       cmp    bl, 3
4471 0000E8C9 776A                <1>       ja     short sysvideo_15_51
4472                              <1>
4473                              <1>       ; BL = 3 = NOT bits in window (ECX, EDX)
4474 0000E8CB 5E                  <1>       pop    esi ; * ; LFB address
4475                              <1>
4476 0000E8CC 39CA                <1>       cmp    edx, ecx ; bottom-right > top-left ?
4477 0000E8CE 7318                <1>       jnb    short sysvideo_15_47 ; window
4478                              <1>
```

```
4479 0000E8D0 09D2              <1>       or     edx, edx
4480 0000E8D2 755C              <1>       jnz    short sysvideo_15_50  ; invalid !
4481                            <1>
4482                            <1>       ; full screen (update) ; edx = 0, ecx > 0
4483 0000E8D4 89C1              <1>       mov    ecx, eax ; LFB video page byte count
4484 0000E8D6 A3[64030300]      <1>       mov    [u.r0], eax
4485 0000E8DB C1E902            <1>       shr    ecx, 2  ; dword 'NOT'
4486                            <1> sysvideo_15_46:
4487 0000E8DE F716              <1>       not    dword [esi]
4488 0000E8E0 AD                <1>       lodsd  ; add esi, 4
4489 0000E8E1 E2FB              <1>       loop   sysvideo_15_46
4490 0000E8E3 E904F0FFFF        <1>       jmp    sysret
4491                            <1>
4492                            <1> sysvideo_15_47:
4493                            <1>       ; 29/12/2020
4494                            <1>       ; fix dx to bp-1 if dx >= bp
4495 0000E8E8 E83C120000        <1>       call   sysvideo_15_200
4496                            <1>       ;
4497                            <1>       ; 29/12/2020
4498 0000E8ED 89F7              <1>       mov    edi, esi
4499 0000E8EF 01C7              <1>       add    edi, eax ; end word of video page + 2
4500 0000E8F1 0FB7C2            <1>       movzx  eax, dx ; bottom right column
4501 0000E8F4 6629C8            <1>       sub    ax, cx  ; - top left column
4502 0000E8F7 7237              <1>       jb      short sysvideo_15_50 ; invalid
4503 0000E8F9 40                <1>         inc  eax ; same column no == 1 column
4504 0000E8FA 50                <1>       push   eax ; pixel count per window row
4505 0000E8FB 52                <1>       push   edx
4506                            <1>       ;movzx ebx, word [ebp+2] ; screen width
4507                            <1>       ; 30/12/2020
4508                            <1>       ; calculate start offset
4509 0000E8FC E833120000        <1>       call   sysvideo_15_202
4510 0000E901 D1E0              <1>       shl    eax, 1 ; pixel pos -> byte pos
4511 0000E903 01C6              <1>       add    esi, eax ; start address
4512 0000E905 59                <1>       pop    ecx ; edx
4513                            <1>       ;mov   eax, ecx
4514                            <1>       ;shr   eax, 16 ; bottom row  ; zero based
4515                            <1>       ;mul   ebx
4516                            <1>       ;mov   dx, cx ; bottom right column ; 0 based
4517                            <1>       ;add   eax, edx
4518                            <1>       ; 30/12/2020
4519                            <1>       ; calculate start offset
4520 0000E906 E82B120000        <1>       call   sysvideo_15_203
4521 0000E90B D1E0              <1>       shl    eax, 1 ; pixel pos -> byte pos
4522 0000E90D 5A                <1>       pop    edx ; pixel count per window row
4523                            <1>       ; eax = the last byte to be converted
4524 0000E90E 01F0              <1>       add    eax, esi
4525 0000E910 39C7              <1>       cmp    edi, eax
4526 0000E912 761C              <1>       jna    short sysvideo_15_50 ; out of video page
4527 0000E914 89C7              <1>       mov    edi, eax ; stop address
4528 0000E916 29D3              <1>       sub    ebx, edx ; screen width - window width
4529 0000E918 D1E3              <1>       shl    ebx, 1 ; pixel count -> byte count
4530 0000E91A D1E2              <1>       shl    edx, 1 ; byte count
4531 0000E91C 4E                <1>       dec    esi
4532                            <1> sysvideo_15_48:
4533 0000E91D 89D1              <1>       mov    ecx, edx ; window width (byte count)
4534                            <1> sysvideo_15_49:
4535 0000E91F 46                <1>       inc    esi
4536 0000E920 F616              <1>       not    byte [esi]
4537 0000E922 E2FB              <1>       loop   sysvideo_15_49
4538 0000E924 0115[64030300]    <1>       add    [u.r0], edx ; 1 pixel = 2 bytes
4539 0000E92A 01DE              <1>       add    esi, ebx ; add difference for next row
4540                            <1>       ;
4541 0000E92C 39FE              <1>       cmp    esi, edi ; stop address (included in loop)
4542 0000E92E 76ED              <1>       jna    short sysvideo_15_48
4543                            <1> sysvideo_15_50:
4544 0000E930 E9B7EFFFFF        <1>       jmp    sysret
4545                            <1>
4546                            <1> sysvideo_15_51:
4547 0000E935 80FB04            <1>       cmp    bl, 4
4548 0000E938 7779              <1>         ja   short sysvideo_15_55
4549                            <1>
4550                            <1>       ; BL = 4 = window copy (system to system)
4551                            <1>
4552                            <1>       ; eax = LFB video page byte count
4553                            <1>       ;      (2 * pixel count)
4554                            <1>
4555 0000E93A 5B                <1>       pop    ebx ; * ; LFB address
4556                            <1>
4557 0000E93B 39DE              <1>       cmp    esi, ebx
4558 0000E93D 72F1              <1>       jb     short sysvideo_15_50
4559                            <1>
4560 0000E93F 39DF              <1>       cmp    edi, ebx
4561 0000E941 72ED              <1>       jb     short sysvideo_15_50
4562 0000E943 01C3              <1>       add    ebx, eax ; **
4563 0000E945 39DE              <1>       cmp    esi, ebx
4564 0000E947 73E7              <1>       jnb    short sysvideo_15_50
4565 0000E949 39DF              <1>       cmp    edi, ebx
4566 0000E94B 73E3              <1>       jnb    short sysvideo_15_50
4567                            <1>
4568 0000E94D 39CA              <1>       cmp    edx, ecx ; bottom-right > top-left ?
4569 0000E94F 7312              <1>       jnb    short sysvideo_15_52 ; window
4570                            <1>
4571 0000E951 09D2              <1>       or     edx, edx
4572 0000E953 75DB              <1>       jnz    short sysvideo_15_50 ; invalid !
4573                            <1>
4574 0000E955 A3[64030300]      <1>       mov    [u.r0], eax
4575 0000E95A 89C1              <1>       mov    ecx, eax ; byte count
4576                            <1>       ; full screen copy ; edx = 0, ecx > 0
4577 0000E95C F3A4              <1>       rep    movsb
4578 0000E95E E989EFFFFF        <1>       jmp    sysret
4579                            <1> sysvideo_15_52:
4580                            <1>       ; 29/12/2020
4581                            <1>       ; fix dx to bp-1 if dx >= bp
4582 0000E963 E8C1110000        <1>       call   sysvideo_15_200
4583                            <1>       ;
```

```
4584                                    <1>      ; 29/12/2020
4585 0000E968 6639CA                    <1>      cmp    dx, cx
4586 0000E96B 72C3                      <1>        jb      short sysvideo_15_50 ; invalid
4587                                    <1>
4588 0000E96D 53                        <1>      push   ebx ; end word of video page + 2
4589 0000E96E 29C3                      <1>      sub    ebx, eax ; **
4590 0000E970 53                        <1>      push   ebx ; LFB start/base address
4591                                    <1>
4592 0000E971 0FB7C2                    <1>      movzx  eax, dx ; bottom right column
4593 0000E974 6629C8                    <1>      sub    ax, cx  ; - top left column
4594 0000E977 40                        <1>      inc    eax ; same column no == 1 column
4595 0000E978 50                        <1>      push   eax ; pixel count per window row
4596 0000E979 52                        <1>      push   edx
4597                                    <1>      ;movzx ebx, word [ebp+2] ; screen width
4598                                    <1>      ; 30/12/2020
4599                                    <1>      ; calculate start offset
4600 0000E97A E8B5110000                <1>      call   sysvideo_15_202
4601 0000E97F D1E0                      <1>      shl    eax, 1 ; pixel pos -> byte pos
4602 0000E981 01C7                      <1>      add    edi, eax ; start address
4603 0000E983 01C6                      <1>      add    esi, eax
4604 0000E985 59                        <1>      pop    ecx ; edx
4605                                    <1>      ; 30/12/2020
4606                                    <1>      ; calculate start offset
4607 0000E986 E8AB110000                <1>      call   sysvideo_15_203
4608 0000E98B D1E0                      <1>      shl    eax, 1 ; pixel pos -> byte pos
4609 0000E98D 5A                        <1>      pop    edx ; pixel count per window row
4610 0000E98E 59                        <1>      pop    ecx ; LFB start/base address
4611 0000E98F 01C8                      <1>      add    eax, ecx ; stop address
4612 0000E991 59                        <1>      pop    ecx ; the last word of LFB + 2
4613 0000E992 39C8                      <1>      cmp    eax, ecx
4614 0000E994 739A                      <1>      jnb    short sysvideo_15_50
4615                                    <1>
4616                                    <1>      ;push  edi ; start address
4617                                    <1>      ;push  eax ; stop address (included)
4618 0000E996 29D3                      <1>      sub    ebx, edx ; screen width - window width
4619 0000E998 D1E3                      <1>      shl    ebx, 1 ; pixel count -> byte count
4620 0000E99A D1E2                      <1>      shl    edx, 1 ; pixel count -> byte count
4621                                    <1> sysvideo_15_53:
4622 0000E99C 89D1                      <1>      mov    ecx, edx
4623 0000E99E F3A4                      <1>      rep    movsb
4624 0000E9A0 0115[64030300]            <1>      add    [u.r0], edx ; 1 pixel = 2 bytes
4625                                    <1>      ;or    ebx, ebx
4626                                    <1>      ;jz    short sysvideo_15_54
4627 0000E9A6 01DF                      <1>      add    edi, ebx ; add difference (for next row)
4628 0000E9A8 01DE                      <1>      add    esi, ebx
4629                                    <1> ;sysvideo_15_54:
4630                                    <1>      ;cmp   edi, [esp] ; stop addr (included in loop)
4631 0000E9AA 39C7                      <1>      cmp    edi, eax
4632 0000E9AC 76EE                      <1>      jna    short sysvideo_15_53
4633                                    <1> sysvideo_15_54:
4634                                    <1>      ;pop   ebx ; stop address
4635                                    <1>      ;pop   edi ; start address
4636 0000E9AE E939EFFFFF                <1>      jmp    sysret
4637                                    <1>
4638                                    <1> sysvideo_15_55:
4639 0000E9B3 80FB05                    <1>      cmp    bl, 5
4640 0000E9B6 775A                      <1>        ja      short sysvideo_15_60
4641                                    <1>
4642                                    <1>      ; BL = 5 = window copy (user to system)
4643                                    <1>
4644                                    <1>      ; eax = LFB video page byte count
4645                                    <1>      ;      (2 * pixel count)
4646                                    <1>
4647 0000E9B8 5F                        <1>      pop    edi ; * ; LFB address
4648                                    <1>
4649                                    <1>      ; esi = user buffer (in user's memory space)
4650 0000E9B9 39CA                      <1>      cmp    edx, ecx ; bottom-right > top-left ?
4651 0000E9BB 7309                      <1>             jnb    short sysvideo_15_56 ; window
4652                                    <1>
4653 0000E9BD 09D2                      <1>      or     edx, edx
4654                                    <1>      ;jnz   sysret  ; invalid !
4655 0000E9BF 75ED                      <1>      jnz    short sysvideo_15_54
4656                                    <1>
4657                                    <1>      ; full screen copy ; edx = 0, ecx > 0
4658 0000E9C1 E9F7FEFFFF                <1>      jmp    sysvideo_15_44 ; full screen copy
4659                                    <1>
4660                                    <1> sysvideo_15_56:
4661                                    <1>      ; 29/12/2020
4662                                    <1>      ; fix dx to bp-1 if dx >= bp
4663 0000E9C6 E85E110000                <1>      call   sysvideo_15_200
4664                                    <1>      ;
4665                                    <1>      ; 29/12/2020
4666 0000E9CB 6639CA                    <1>      cmp    dx, cx
4667 0000E9CE 72DE                      <1>      jb     short sysvideo_15_54
4668                                    <1>
4669                                    <1>      ; 01/01/2021
4670 0000E9D0 01F8                      <1>      add    eax, edi
4671 0000E9D2 50                        <1>      push   eax ; the last word of LFB + 2
4672 0000E9D3 57                        <1>      push   edi ; LFB start/base address
4673                                    <1>
4674 0000E9D4 0FB7C2                    <1>      movzx  eax, dx ; bottom right column
4675 0000E9D7 6629C8                    <1>      sub    ax, cx  ; - top left column
4676 0000E9DA 40                        <1>      inc    eax ; same column no == 1 column
4677 0000E9DB 50                        <1>      push   eax ; pixel count per window row
4678                                    <1>
4679 0000E9DC 52                        <1>      push   edx
4680                                    <1>      ;movzx ebx, word [ebp+2] ; screen width
4681                                    <1>      ; 30/12/2020
4682                                    <1>      ; calculate start offset
4683 0000E9DD E852110000                <1>      call   sysvideo_15_202
4684 0000E9E2 D1E0                      <1>      shl    eax, 1 ; pixel pos -> byte pos
4685 0000E9E4 01C7                      <1>      add    edi, eax ; start address
4686 0000E9E6 59                        <1>      pop    ecx ; edx
4687                                    <1>      ; 30/12/2020
4688                                    <1>      ; calculate end offset
```

```
4689 0000E9E7 E84A110000          <1>       call   sysvideo_15_203
4690 0000E9EC D1E0                <1>       shl    eax, 1 ; pixel pos -> byte pos
4691 0000E9EE 5A                  <1>       pop    edx ; pixel count per window row
4692 0000E9EF 59                  <1>       pop    ecx ; LFB start/base address
4693 0000E9F0 01C8                <1>       add    eax, ecx ; stop address
4694 0000E9F2 59                  <1>       pop    ecx ; the last word of LFB + 2
4695 0000E9F3 39C8                <1>       cmp    eax, ecx
4696                              <1>       ;jnb   sysret
4697 0000E9F5 73B7                <1>       jnb    short sysvideo_15_54
4698                              <1>
4699                              <1>       ; 29/12/2020
4700 0000E9F7 D1E3                <1>       shl    ebx, 1 ; from pixel count to byte count
4701 0000E9F9 D1E2                <1>       shl    edx, 1 ; pixel count -> byte count
4702                              <1>       ;push  edi ; start address
4703 0000E9FB 50                  <1>       push   eax ; stop address (included)
4704                              <1> sysvideo_15_57:
4705 0000E9FC 89D1                <1>       mov    ecx, edx ; byte count
4706                              <1>       ; user to system video/display page window transfer
4707                              <1>       ; esi =      user buffer
4708 0000E9FE E80B290000          <1>       call   transfer_from_user_buffer ; fast transfer
4709 0000EA03 7207                <1>       jc     short sysvideo_15_58
4710                              <1>              ; ecx = actual transfer count (= ecx input)
4711                              <1>       ;add   [u.r0], ecx
4712                              <1>       ;add   edi, ebx ; next row in video memory window
4713                              <1>       ;add   esi, ecx ; next window row in user's buffer
4714                              <1>       ;cmp   edi, [esp] ; stop addr (included in loop)
4715                              <1>       ; 30/12/2020
4716 0000EA05 E839110000          <1>       call   sysvideo_15_204
4717 0000EA0A 76F0                <1>       jna    short sysvideo_15_57
4718                              <1> sysvideo_15_58:
4719 0000EA0C 5B                  <1>       pop    ebx ; stop address
4720                              <1>       ;pop   edi ; start address
4721                              <1> sysvideo_15_59:
4722 0000EA0D E9DAEEFFFF          <1>       jmp    sysret
4723                              <1>
4724                              <1> sysvideo_15_60:
4725 0000EA12 80FB06              <1>       cmp    bl, 6
4726 0000EA15 775A                <1>       ja     short sysvideo_15_63
4727                              <1>
4728                              <1>       ; BL = 6 = window copy (system to user)
4729                              <1>
4730                              <1>       ; eax = LFB video page byte count
4731                              <1>       ;      (2 * pixel count)
4732                              <1>
4733 0000EA17 89F7                <1>       mov    edi, esi ; user buffer
4734                              <1>
4735 0000EA19 5E                  <1>       pop    esi ; * ; LFB address
4736                              <1>
4737                              <1>       ; edi = user buffer (in user's memory space)
4738 0000EA1A 39CA                <1>       cmp    edx, ecx ; bottom-right > top-left ?
4739 0000EA1C 7309                <1>         jnb     short sysvideo_15_61
4740                              <1>
4741 0000EA1E 21D2                <1>       and    edx, edx
4742                              <1>       ;jnz   sysret ; invalid !
4743 0000EA20 75EB                <1>       jnz    short sysvideo_15_59
4744                              <1>
4745                              <1>       ; full screen copy  ; edx = 0, ecx > 0
4746 0000EA22 E97FFEFFFF          <1>       jmp    sysvideo_15_41
4747                              <1>
4748                              <1> sysvideo_15_61:
4749                              <1>       ; 29/12/2020
4750                              <1>       ; fix dx to bp-1 if dx >= bp
4751 0000EA27 E8FD100000          <1>       call   sysvideo_15_200
4752                              <1>       ;
4753                              <1>       ; 29/12/2020
4754 0000EA2C 6639CA              <1>       cmp    dx, cx
4755 0000EA2F 72DC                <1>       jb     short sysvideo_15_59
4756                              <1>
4757 0000EA31 89F3                <1>       mov    ebx, esi ; LFB address
4758 0000EA33 01D8                <1>       add    eax, ebx
4759 0000EA35 50                  <1>       push   eax ; the last word of LFB + 2
4760 0000EA36 53                  <1>       push   ebx ; LFB start/base address
4761                              <1>
4762 0000EA37 0FB7C2              <1>       movzx  eax, dx ; bottom right column
4763 0000EA3A 6629C8              <1>       sub    ax, cx  ; - top left column
4764 0000EA3D 40                  <1>       inc    eax ; same column no == 1 column
4765 0000EA3E 50                  <1>       push   eax ; pixel count per window row
4766                              <1>
4767 0000EA3F 52                  <1>       push   edx
4768                              <1>       ;movzx ebx, word [ebp+2] ; screen width
4769                              <1>       ; 30/12/2020
4770                              <1>       ; calculate start offset
4771 0000EA40 E8EF100000          <1>       call   sysvideo_15_202
4772 0000EA45 D1E0                <1>       shl    eax, 1 ; pixel pos -> byte pos
4773 0000EA47 01C6                <1>       add    esi, eax ; start address
4774 0000EA49 59                  <1>       pop    ecx ; edx
4775                              <1>       ; 30/12/2020
4776                              <1>       ; calculate end offset
4777 0000EA4A E8E7100000          <1>       call   sysvideo_15_203
4778 0000EA4F D1E0                <1>       shl    eax, 1 ; pixel pos -> byte pos
4779 0000EA51 5A                  <1>       pop    edx ; pixel count per window row
4780 0000EA52 59                  <1>       pop    ecx ; LFB start/base address
4781 0000EA53 01C8                <1>       add    eax, ecx ; stop address
4782 0000EA55 59                  <1>       pop    ecx ; the last word of LFB + 2
4783 0000EA56 39C8                <1>       cmp    eax, ecx
4784                              <1>       ;jnb   sysret
4785 0000EA58 73B3                <1>       jnb    short sysvideo_15_59
4786                              <1>
4787                              <1>       ; 29/12/2020
4788 0000EA5A D1E3                <1>       shl    ebx, 1 ; from pixel count to byte count
4789 0000EA5C D1E2                <1>       shl    edx, 1 ; pixel count -> byte count
4790                              <1>       ;push  esi ; start address
4791 0000EA5E 50                  <1>       push   eax ; stop address (included)
4792                              <1> sysvideo_15_62:
4793 0000EA5F 89D1                <1>       mov    ecx, edx ; byte count
```

```
                                <1>       ; user to system video/display page window transfer
                                <1>       ; esi =      user buffer
0000EA61 E85E280000             <1>       call   transfer_to_user_buffer ; fast transfer
                                <1>       ;jc    short sysvideo_15_62_0
0000EA66 72A4                   <1>       jc     short sysvideo_15_58
                                <1>       ;add   [u.r0], ecx
                                <1>       ;add   edi, ebx ; next row in video memory window
                                <1>       ;add   esi, ecx ; next window row in user's buffer
                                <1>       ;cmp   edi, [esp] ; stop addr (included in loop)
                                <1>       ; 30/12/2020
0000EA68 E8D6100000             <1>       call   sysvideo_15_204
0000EA6D 76F0                   <1>       jna    short sysvideo_15_62
0000EA6F EB9B                   <1>       jmp    short sysvideo_15_58
                                <1>
                                <1> sysvideo_15_63:
                                <1>       ; bl = 7, 8, 9, 10, 11, 12, 13
                                <1>       ; (bl > 13 is invalid for current kernel version)
                                <1>
                                <1>       ;cmp   bl, 13
                                <1>       ;ja    sysvideo_15_58
                                <1>
                                <1>       ; 01/01/2021
0000EA71 80FB0E                 <1>       cmp    bl, 14
0000EA74 0F83C7060000           <1>       jnb    sysvideo_15_151
                                <1>
                                <1>       ; eax = LFB video page size (w*h*2) in bytes
                                <1>
0000EA7A 5E                     <1>       pop    esi ; * ; LFB base/start address
0000EA7B 91                     <1>       xchg   eax, ecx ; ecx = byte count
                                <1>                       ; ax = word for and/or/xor
0000EA7C 890D[64030300]         <1>       mov    [u.r0], ecx ; return with
                                <1>                         ; byte count of display page
                                <1>                         ; (eax > 0 -> OK)
0000EA82 D1E9                   <1>       shr    ecx, 1 ; word count
                                <1>
0000EA84 80FB0A                 <1>       cmp    bl, 10
0000EA87 722B                   <1>       jb     short sysvideo_15_67 ; 7, 8, 9
0000EA89 740E                   <1>       je     short sysvideo_15_64 ; 10
0000EA8B 80FB0C                 <1>       cmp    bl, 12
0000EA8E 7212                   <1>       jb     short sysvideo_15_65 ; 11
0000EA90 7419                   <1>       je     short sysvideo_15_66 ; 12
                                <1>
                                <1>       ; BL = 13 = NOT display page words
0000EA92 D1E9                   <1>       shr    ecx, 1 ; dword count
                                <1>       ; 27/12/2020
0000EA94 E945FEFFFF             <1>       jmp    sysvideo_15_46
                                <1>
                                <1>       ; BL = 10 = INC display page words
                                <1> sysvideo_15_64:
0000EA99 66FF06                 <1>       inc    word [esi]
0000EA9C 46                     <1>       inc    esi
0000EA9D 46                     <1>       inc    esi
0000EA9E E2F9                   <1>       loop   sysvideo_15_64
0000EAA0 EB38                   <1>       jmp    short sysvideo_15_71
                                <1>
                                <1>       ; BL = 11 = DEC display page words
                                <1> sysvideo_15_65:
0000EAA2 66FF0E                 <1>       dec    word [esi]
0000EAA5 46                     <1>       inc    esi
0000EAA6 46                     <1>       inc    esi
0000EAA7 E2F9                   <1>       loop   sysvideo_15_65
0000EAA9 EB2F                   <1>       jmp    short sysvideo_15_71
                                <1>
                                <1>       ; BL = 12 = NEG display page words
                                <1> sysvideo_15_66:
0000EAAB 66F71E                 <1>       neg    word [esi]
0000EAAE 46                     <1>       inc    esi
0000EAAF 46                     <1>       inc    esi
0000EAB0 E2F9                   <1>       loop   sysvideo_15_66
0000EAB2 EB26                   <1>       jmp    short sysvideo_15_71
                                <1>
                                <1> sysvideo_15_67:
                                <1>       ; bl = 7, 8, 9
                                <1>       ; ecx = byte (pixel) count
0000EAB4 C1E902                 <1>       shr    ecx, 2 ; dword count
0000EAB7 6689C2                 <1>       mov    dx, ax
0000EABA C1E210                 <1>       shl    edx, 16
0000EABD 6689C2                 <1>       mov    dx, ax
                                <1>                ; edx = 2 words for and/or/xor
                                <1>
0000EAC0 80FB08                 <1>       cmp    bl, 8
0000EAC3 7409                   <1>       je     short sysvideo_15_69
0000EAC5 770E                   <1>       ja     short sysvideo_15_70
                                <1>
                                <1>       ; BL = 7 = AND display page words with CX
                                <1> sysvideo_15_68:
0000EAC7 2116                   <1>       and    [esi], edx
0000EAC9 AD                     <1>       lodsd
0000EACA E2FB                   <1>       loop   sysvideo_15_68
                                <1>       ;jmp   sysret
0000EACC EB0C                   <1>       jmp    short sysvideo_15_71
                                <1>
                                <1>       ; BL = 8 = OR display page words with CX
                                <1> sysvideo_15_69:
0000EACE 0916                   <1>       or     [esi], edx
0000EAD0 AD                     <1>       lodsd
0000EAD1 E2FB                   <1>       loop   sysvideo_15_69
                                <1>       ;jmp   sysret
0000EAD3 EB05                   <1>       jmp    short sysvideo_15_71
                                <1>
                                <1>       ; BL = 9 = XOR display page words with CX
                                <1> sysvideo_15_70:
0000EAD5 3116                   <1>       xor    [esi], edx
0000EAD7 AD                     <1>       lodsd
0000EAD8 E2FB                   <1>       loop   sysvideo_15_70
```

```
                                <1> sysvideo_15_71:
0000EADA E90DEEFFFF              <1>       jmp    sysret
                                <1>
                                <1> sysvideo_15_72:
                                <1>       ; 27/12/2020
                                <1>       ; edx = 0
                                <1>
                                <1>       ; bh = bits per pixel (bpp) = 32
                                <1>
                                <1>       ; eax = linear frame buffer (VIDEO PAGE) pixel count
                                <1>       ;       (a quarter of video page size in bytes)
                                <1>
0000EADF C1E002                 <1>       shl    eax, 2 ; byte count = 4 * pixel count
                                <1>
                                <1>       ; sub function for sysvideo function bh = 2
0000EAE2 20DB                   <1>       and    bl, bl ; 0 ?
0000EAE4 7511                   <1>       jnz    short sysvideo_15_73 ; no
                                <1>
                                <1>       ; BL = 0 = Fill color (color in ECX)
                                <1>
0000EAE6 5F                     <1>       pop    edi ; * ; LFB address
                                <1>
0000EAE7 A3[64030300]           <1>       mov    [u.r0], eax ; return value to user
                                <1>
0000EAEC 91                     <1>       xchg   eax, ecx ; eax = color, ecx = pixel count
                                <1>
                                <1>       ; 32 bit pixels
0000EAED C1E902                 <1>       shr    ecx, 2
                                <1>       ; ecx = pixel count
                                <1>
0000EAF0 F3AB                   <1>       rep    stosd
                                <1>
                                <1>       ; [u.r0] = transfer (fill) count in bytes
0000EAF2 E9F5EDFFFF             <1>       jmp    sysret
                                <1>
                                <1> sysvideo_15_73:
                                <1>       ; 27/12/2020
                                <1>
                                <1>       ; eax = linear frame buffer (VIDEO PAGE) byte count
                                <1>       ;       (4 * pixel count)
                                <1>
0000EAF7 80FB02                 <1>       cmp    bl, 2
0000EAFA 7725                   <1>       ja     short sysvideo_15_78
0000EAFC 7217                   <1>       jb     short sysvideo_15_76
                                <1>
                                <1>       ; BL = 2 = system to user video/display page transfer
0000EAFE 5E                     <1>       pop    esi ; * ; LFB address
0000EAFF 89CF                   <1>       mov    edi, ecx ; user's buffer
                                <1> sysvideo_15_74:
0000EB01 89C1                   <1>       mov    ecx, eax ; LFB video page byte count
0000EB03 E8BC270000             <1>       call   transfer_to_user_buffer ; fast transfer
                                <1> sysvideo_15_75:
                                <1>       ;jc    sysret ; [u.r0] = 0
0000EB08 72D0                   <1>       jc     short sysvideo_15_71
0000EB0A 890D[64030300]         <1>       mov    [u.r0], ecx ; actual transfer count
0000EB10 E9D7EDFFFF             <1>       jmp    sysret
                                <1>
                                <1> sysvideo_15_76:
                                <1>       ; BL = 1 = user to system video/display page transfer
0000EB15 5F                     <1>       pop    edi ; * ; LFB address
0000EB16 89CE                   <1>       mov    esi, ecx ; user's buffer
                                <1> sysvideo_15_77:
0000EB18 89C1                   <1>       mov    ecx, eax ; LFB video page byte count
0000EB1A E8EF270000             <1>       call   transfer_from_user_buffer ; fast transfer
0000EB1F EBE7                   <1>       jmp    short sysvideo_15_75
                                <1>
                                <1> sysvideo_15_78:
0000EB21 80FB03                 <1>       cmp    bl, 3
0000EB24 7776                   <1>       ja     short sysvideo_15_84
                                <1>
                                <1>       ; BL = 3 = NOT bits in window (ECX, EDX)
0000EB26 5E                     <1>       pop    esi ; * ; LFB address
                                <1>
0000EB27 39CA                   <1>       cmp    edx, ecx ; bottom-right > top-left ?
0000EB29 7318                   <1>       jnb    short sysvideo_15_80 ; window
                                <1>
0000EB2B 09D2                   <1>       or     edx, edx
0000EB2D 7568                   <1>       jnz    short sysvideo_15_83  ; invalid !
                                <1>
                                <1>       ; full screen (update) ; edx = 0, ecx > 0
0000EB2F 89C1                   <1>       mov    ecx, eax ; LFB video page byte count
0000EB31 A3[64030300]           <1>       mov    [u.r0], eax
0000EB36 C1E902                 <1>       shr    ecx, 2  ; dword 'NOT'
                                <1> sysvideo_15_79:
0000EB39 F716                   <1>       not    dword [esi]
0000EB3B AD                     <1>       lodsd  ; add esi, 4
0000EB3C E2FB                   <1>       loop   sysvideo_15_79
0000EB3E E9A9EDFFFF             <1>       jmp    sysret
                                <1>
                                <1> sysvideo_15_80:
                                <1>       ; 29/12/2020
                                <1>       ; fix dx to bp-1 if dx >= bp
0000EB43 E8E10F0000             <1>       call   sysvideo_15_200
                                <1>       ;
                                <1>       ; 29/12/2020
0000EB48 6639CA                 <1>       cmp    dx, cx
0000EB4B 724A                   <1>         jb      short sysvideo_15_83 ; invalid
                                <1>
0000EB4D 89F7                   <1>       mov    edi, esi
0000EB4F 01C7                   <1>       add    edi, eax ; end dword of video page + 4
0000EB51 0FB7C2                 <1>       movzx  eax, dx ; bottom right column
0000EB54 6629C8                 <1>       sub    ax, cx  ; - top left column
0000EB57 40                     <1>       inc    eax ; same column no == 1 column
0000EB58 50                     <1>       push   eax ; pixel count per window row
0000EB59 52                     <1>       push   edx
```

```
5004                                    <1>       ;movzx ebx, word [ebp+2] ; screen width
5005                                    <1>       ; 30/12/2020
5006                                    <1>       ; calculate start offset
5007 0000EB5A E8D50F0000                <1>       call   sysvideo_15_202
5008 0000EB5F C1E002                    <1>       shl    eax, 2 ; pixel pos -> byte pos
5009 0000EB62 01C6                      <1>       add    esi, eax ; start address
5010 0000EB64 59                        <1>       pop    ecx ; edx
5011                                    <1>       ; 30/12/2020
5012                                    <1>       ; calculate start offset
5013 0000EB65 E8CC0F0000                <1>       call   sysvideo_15_203
5014 0000EB6A C1E002                    <1>       shl    eax, 2 ; pixel pos -> byte pos
5015 0000EB6D 5A                        <1>       pop    edx ; pixel count per window row
5016                                    <1>       ; eax = the last byte to be converted
5017 0000EB6E 01F0                      <1>       add    eax, esi
5018 0000EB70 39C7                      <1>       cmp    edi, eax
5019 0000EB72 7623                      <1>       jna    short sysvideo_15_83 ; out of video page
5020 0000EB74 89C7                      <1>       mov    edi, eax ; stop address
5021                                    <1>
5022 0000EB76 29D3                      <1>       sub    ebx, edx
5023 0000EB78 C1E302                    <1>       shl    ebx, 2 ; pixel count -> byte count
5024 0000EB7B C1E202                    <1>       shl    edx, 2 ; byte count
5025 0000EB7E 83EE04                    <1>       sub    esi, 4
5026                                    <1> sysvideo_15_81:
5027 0000EB81 89D1                      <1>       mov    ecx, edx ; window width (byte count)
5028 0000EB83 C1E902                    <1>       shr    ecx, 2
5029                                    <1> sysvideo_15_82:
5030 0000EB86 AD                        <1>       lodsd
5031 0000EB87 F716                      <1>       not    dword [esi]
5032 0000EB89 E2FB                      <1>       loop   sysvideo_15_82
5033 0000EB8B 0115[64030300]            <1>       add    [u.r0], edx ; 1 pixel = 4 bytes
5034 0000EB91 01DE                      <1>       add    esi, ebx ; add difference for next row
5035                                    <1>       ;
5036 0000EB93 39FE                      <1>       cmp    esi, edi ; stop address (included in loop)
5037 0000EB95 76EA                      <1>       jna    short sysvideo_15_81
5038                                    <1> sysvideo_15_83:
5039 0000EB97 E950EDFFFF                <1>       jmp    sysret
5040                                    <1>
5041                                    <1> sysvideo_15_84:
5042 0000EB9C 80FB04                    <1>       cmp    bl, 4
5043 0000EB9F 0F8783000000              <1>         ja   sysvideo_15_88
5044                                    <1>
5045                                    <1>       ; BL = 4 = window copy (system to system)
5046                                    <1>
5047                                    <1>       ; eax = LFB video page byte count
5048                                    <1>       ;      (4 * pixel count)
5049                                    <1>
5050 0000EBA5 5B                        <1>       pop    ebx ; * ; LFB address
5051                                    <1>
5052 0000EBA6 39DE                      <1>       cmp    esi, ebx
5053 0000EBA8 72ED                      <1>       jb     short sysvideo_15_83
5054                                    <1>
5055 0000EBAA 39DF                      <1>       cmp    edi, ebx
5056 0000EBAC 72E9                      <1>       jb     short sysvideo_15_83
5057 0000EBAE 01C3                      <1>       add    ebx, eax ; **
5058 0000EBB0 39DE                      <1>       cmp    esi, ebx
5059 0000EBB2 73E3                      <1>       jnb    short sysvideo_15_83
5060 0000EBB4 39DF                      <1>       cmp    edi, ebx
5061 0000EBB6 73DF                      <1>       jnb    short sysvideo_15_83
5062                                    <1>
5063 0000EBB8 39CA                      <1>       cmp    edx, ecx ; bottom-right > top-left ?
5064 0000EBBA 7315                      <1>       jnb    short sysvideo_15_85 ; window
5065                                    <1>
5066 0000EBBC 09D2                      <1>       or     edx, edx
5067 0000EBBE 75D7                      <1>       jnz    short sysvideo_15_83 ; invalid !
5068                                    <1>
5069 0000EBC0 A3[64030300]              <1>       mov    [u.r0], eax
5070 0000EBC5 89C1                      <1>       mov    ecx, eax ; byte count
5071 0000EBC7 C1E902                    <1>       shr    ecx, 2
5072                                    <1>       ; full screen copy ; edx = 0, ecx > 0
5073 0000EBCA F3A5                      <1>       rep    movsd
5074 0000EBCC E91BEDFFFF                <1>       jmp    sysret
5075                                    <1> sysvideo_15_85:
5076                                    <1>       ; 29/12/2020
5077                                    <1>       ; fix dx to bp-1 if dx >= bp
5078 0000EBD1 E8530F0000                <1>       call   sysvideo_15_200
5079                                    <1>       ;
5080                                    <1>       ; 29/12/2020
5081 0000EBD6 6639CA                    <1>       cmp    dx, cx
5082 0000EBD9 72BC                      <1>         jb      short sysvideo_15_83 ; invalid
5083                                    <1>
5084 0000EBDB 53                        <1>       push   ebx ; end dword of video page + 4
5085 0000EBDC 29C3                      <1>       sub    ebx, eax ; **
5086 0000EBDE 53                        <1>       push   ebx ; LFB start/base address
5087                                    <1>
5088 0000EBDF 0FB7C2                    <1>       movzx  eax, dx ; bottom right column
5089 0000EBE2 6629C8                    <1>       sub    ax, cx  ; - top left column
5090 0000EBE5 40                        <1>       inc    eax ; same column no == 1 column
5091 0000EBE6 50                        <1>       push   eax ; pixel count per window row
5092 0000EBE7 52                        <1>       push   edx
5093                                    <1>       ;movzx ebx, word [ebp+2] ; screen width
5094                                    <1>       ; 30/12/2020
5095                                    <1>       ; calculate start offset
5096 0000EBE8 E8470F0000                <1>       call   sysvideo_15_202
5097 0000EBED C1E002                    <1>       shl    eax, 2 ; pixel pos -> byte pos
5098 0000EBF0 01C7                      <1>       add    edi, eax ; start address
5099 0000EBF2 01C6                      <1>       add    esi, eax
5100 0000EBF4 59                        <1>       pop    ecx ; edx
5101                                    <1>       ; 30/12/2020
5102                                    <1>       ; calculate start offset
5103 0000EBF5 E83C0F0000                <1>       call   sysvideo_15_203
5104 0000EBFA C1E002                    <1>       shl    eax, 2 ; pixel pos -> byte pos
5105 0000EBFD 5A                        <1>       pop    edx ; pixel count per window row
5106 0000EBFE 59                        <1>       pop    ecx ; LFB start/base address
5107 0000EBFF 01C8                      <1>       add    eax, ecx ; stop address
5108 0000EC01 59                        <1>       pop    ecx ; the last dword of LFB + 4
```

```
5109 0000EC02 39C8                <1>       cmp    eax, ecx
5110 0000EC04 7391                <1>       jnb    short sysvideo_15_83
5111                              <1>
5112                              <1>       ;push  edi ; start address
5113                              <1>       ;push  eax ; stop address (included)
5114 0000EC06 29D3                <1>       sub    ebx, edx ; screen width - window width
5115 0000EC08 C1E302              <1>       shl    ebx, 2 ; pixel count -> byte count
5116 0000EC0B C1E202              <1>       shl    edx, 2 ; pixel count -> byte count
5117                              <1> sysvideo_15_86:
5118 0000EC0E 89D1                <1>       mov    ecx, edx
5119 0000EC10 C1E902              <1>       shr    ecx, 2
5120 0000EC13 F3A5                <1>       rep    movsd
5121 0000EC15 0115[64030300]      <1>       add    [u.r0], edx ; 1 pixel = 4 bytes
5122                              <1>       ;or    ebx, ebx
5123                              <1>       ;jz    short sysvideo_15_87
5124 0000EC1B 01DF                <1>       add    edi, ebx ; add difference (for next row)
5125 0000EC1D 01DE                <1>       add    esi, ebx
5126                              <1> ;sysvideo_15_87:
5127                              <1>       ;cmp   edi, [esp] ; stop addr (included in loop)
5128 0000EC1F 39C7                <1>       cmp    edi, eax
5129 0000EC21 76EB                <1>       jna    short sysvideo_15_86
5130                              <1> sysvideo_15_87:
5131                              <1>       ;pop   ebx ; stop address
5132                              <1>       ;pop   edi ; start address
5133 0000EC23 E9C4ECFFFF          <1>       jmp    sysret
5134                              <1>
5135                              <1> sysvideo_15_88:
5136 0000EC28 80FB05              <1>       cmp    bl, 5
5137 0000EC2B 775E                <1>         ja      short sysvideo_15_93
5138                              <1>
5139                              <1>       ; BL = 5 = window copy (user to system)
5140                              <1>
5141                              <1>       ; eax = LFB video page byte count
5142                              <1>       ;      (4 * pixel count)
5143                              <1>
5144 0000EC2D 5F                  <1>       pop    edi ; * ; LFB address
5145                              <1>
5146                              <1>       ; esi = user buffer (in user's memory space)
5147 0000EC2E 39CA                <1>       cmp    edx, ecx ; bottom-right > top-left ?
5148 0000EC30 7309                <1>              jnb    short sysvideo_15_89 ; window
5149                              <1>
5150 0000EC32 09D2                <1>       or     edx, edx
5151                              <1>       ;jnz   sysret  ; invalid !
5152 0000EC34 75ED                <1>       jnz    short sysvideo_15_87
5153                              <1>
5154                              <1>       ; full screen copy ; edx = 0, ecx > 0
5155 0000EC36 E9DDFEFFFF          <1>       jmp    sysvideo_15_77 ; full screen copy
5156                              <1>
5157                              <1> sysvideo_15_89:
5158                              <1>       ; 29/12/2020
5159                              <1>       ; fix dx to bp-1 if dx >= bp
5160 0000EC3B E8E90E0000          <1>       call   sysvideo_15_200
5161                              <1>       ;
5162                              <1>       ; 29/12/2020
5163 0000EC40 6639CA              <1>       cmp    dx, cx
5164 0000EC43 72DE                <1>       jb     short sysvideo_15_87
5165                              <1>
5166                              <1>       ; 01/01/2021
5167 0000EC45 01F8                <1>       add    eax, edi
5168 0000EC47 50                  <1>       push   eax ; the last dword of LFB + 4
5169 0000EC48 57                  <1>       push   edi ; LFB start/base address
5170                              <1>
5171 0000EC49 0FB7C2              <1>       movzx  eax, dx ; bottom right column
5172 0000EC4C 6629C8              <1>       sub    ax, cx  ; - top left column
5173 0000EC4F 40                  <1>       inc    eax ; same column no == 1 column
5174 0000EC50 50                  <1>       push   eax ; pixel count per window row
5175                              <1>
5176 0000EC51 52                  <1>       push   edx
5177                              <1>       ; 30/12/2020
5178                              <1>       ; calculate start offset
5179 0000EC52 E8DD0E0000          <1>       call   sysvideo_15_202
5180 0000EC57 C1E002              <1>       shl    eax, 2 ; pixel pos -> byte pos
5181 0000EC5A 01C7                <1>       add    edi, eax ; start address
5182 0000EC5C 59                  <1>       pop    ecx ; edx
5183                              <1>       ; 30/12/2020
5184                              <1>       ; calculate end offset
5185 0000EC5D E8D40E0000          <1>       call   sysvideo_15_203
5186 0000EC62 C1E002              <1>       shl    eax, 2 ; pixel pos -> byte pos
5187 0000EC65 5A                  <1>       pop    edx ; pixel count per window row
5188 0000EC66 59                  <1>       pop    ecx ; LFB start/base address
5189 0000EC67 01C8                <1>       add    eax, ecx ; stop address
5190 0000EC69 59                  <1>       pop    ecx ; the last dword of LFB + 4
5191 0000EC6A 39C8                <1>       cmp    eax, ecx
5192                              <1>       ;jnb   sysret
5193 0000EC6C 73B5                <1>       jnb    short sysvideo_15_87
5194                              <1>
5195                              <1>       ; 29/12/220
5196 0000EC6E C1E302              <1>       shl    ebx, 2 ; from pixel count to byte count
5197 0000EC71 C1E202              <1>       shl    edx, 2 ; pixel count -> byte count
5198                              <1>       ;push  edi ; start address
5199 0000EC74 50                  <1>       push   eax ; stop address (included)
5200                              <1> sysvideo_15_90:
5201 0000EC75 89D1                <1>       mov    ecx, edx ; byte count
5202                              <1>       ; user to system video/display page window transfer
5203                              <1>       ; esi =      user buffer
5204 0000EC77 E892260000          <1>       call   transfer_from_user_buffer ; fast transfer
5205 0000EC7C 7207                <1>       jc     short sysvideo_15_91
5206                              <1>              ; ecx = actual transfer count (= ecx input)
5207                              <1>       ;add   [u.r0], ecx
5208                              <1>       ;add   edi, ebx ; next row in video memory window
5209                              <1>       ;add   esi, ecx ; next window row in user's buffer
5210                              <1>       ;cmp   edi, [esp] ; stop addr (included in loop)
5211                              <1>       ; 30/12/2020
5212 0000EC7E E8C00E0000          <1>       call   sysvideo_15_204
5213 0000EC83 76F0                <1>       jna    short sysvideo_15_90
```

```
                            <1> sysvideo_15_91:
0000EC85 5B                 <1>         pop    ebx ; stop address
                            <1>         ;pop   edi ; start address
                            <1> sysvideo_15_92:
0000EC86 E961ECFFFF         <1>         jmp    sysret
                            <1>
                            <1> sysvideo_15_93:
0000EC8B 80FB06             <1>         cmp    bl, 6
0000EC8E 775E               <1>         ja     short sysvideo_15_96
                            <1>
                            <1>         ; BL = 6 = window copy (system to user)
                            <1>
                            <1>         ; eax = LFB video page byte count
                            <1>         ;       (4 * pixel count)
                            <1>
0000EC90 89F7               <1>         mov    edi, esi ; user buffer
                            <1>
0000EC92 5E                 <1>         pop    esi ; * ; LFB address
                            <1>
                            <1>         ; edi = user buffer (in user's memory space)
0000EC93 39CA               <1>         cmp    edx, ecx ; bottom-right > top-left ?
0000EC95 7309               <1>           jnb     short sysvideo_15_94
                            <1>
0000EC97 21D2               <1>         and    edx, edx
                            <1>         ;jnz   sysret ; invalid !
0000EC99 75EB               <1>         jnz    short sysvideo_15_92
                            <1>
                            <1>         ; full screen copy  ; edx = 0, ecx > 0
0000EC9B E961FEFFFF         <1>         jmp    sysvideo_15_74
                            <1>
                            <1> sysvideo_15_94:
                            <1>         ; 29/12/2020
                            <1>         ; fix dx to bp-1 if dx >= bp
0000ECA0 E8840E0000         <1>         call   sysvideo_15_200
                            <1>         ;
                            <1>         ; 29/12/2020
0000ECA5 6639CA             <1>         cmp    dx, cx
0000ECA8 72DC               <1>         jb     short sysvideo_15_92
                            <1>
0000ECAA 89F3               <1>         mov    ebx, esi ; LFB address
0000ECAC 01D8               <1>         add    eax, ebx
0000ECAE 50                 <1>         push   eax ; the last dword of LFB + 4
0000ECAF 53                 <1>         push   ebx ; LFB start/base address
                            <1>
0000ECB0 0FB7C2             <1>         movzx  eax, dx ; bottom right column
0000ECB3 6629C8             <1>         sub    ax, cx  ; - top left column
0000ECB6 40                 <1>         inc    eax ; same column no == 1 column
0000ECB7 50                 <1>         push   eax ; pixel count per window row
                            <1>
0000ECB8 52                 <1>         push   edx
                            <1>         ;movzx ebx, word [ebp+2] ; screen width
                            <1>         ; 30/12/2020
                            <1>         ; calculate start offset
0000ECB9 E8760E0000         <1>         call   sysvideo_15_202
0000ECBE C1E002             <1>         shl    eax, 2 ; pixel pos -> byte pos
0000ECC1 01C6               <1>         add    esi, eax ; start address
0000ECC3 59                 <1>         pop    ecx ; edx
                            <1>         ; 30/12/2020
                            <1>         ; calculate end offset
0000ECC4 E86D0E0000         <1>         call   sysvideo_15_203
0000ECC9 C1E002             <1>         shl    eax, 2 ; pixel pos -> byte pos
0000ECCC 5A                 <1>         pop    edx ; pixel count per window row
0000ECCD 59                 <1>         pop    ecx ; LFB start/base address
0000ECCE 01C8               <1>         add    eax, ecx ; stop address
0000ECD0 59                 <1>         pop    ecx ; the last dword of LFB + 4
0000ECD1 39C8               <1>         cmp    eax, ecx
                            <1>         ;jnb   sysret
0000ECD3 73B1               <1>         jnb    short sysvideo_15_92
                            <1>
                            <1>         ; 29/12/2020
0000ECD5 C1E302             <1>         shl    ebx, 2 ; from pixel count to byte count
0000ECD8 C1E202             <1>         shl    edx, 2 ; pixel count -> byte count
                            <1>         ;push  esi ; start address
0000ECDB 50                 <1>         push   eax ; stop address (included)
                            <1> sysvideo_15_95:
0000ECDC 89D1               <1>         mov    ecx, edx ; byte count
                            <1>         ; user to system video/display page window transfer
                            <1>         ; esi =      user buffer
0000ECDE E8E1250000         <1>         call   transfer_to_user_buffer ; fast transfer
                            <1>         ;jc    short sysvideo_15_95_0
0000ECE3 72A0               <1>         jc     short sysvideo_15_91
                            <1>         ;add   [u.r0], ecx
                            <1>         ;add   edi, ebx ; next row in video memory window
                            <1>         ;add   esi, ecx ; next window row in user's buffer
                            <1>         ;cmp   edi, [esp] ; stop addr (included in loop)
                            <1>         ; 30/12/2020
0000ECE5 E8590E0000         <1>         call   sysvideo_15_204
0000ECEA 76F0               <1>         jna    short sysvideo_15_95
0000ECEC EB97               <1>         jmp    short sysvideo_15_91
                            <1>
                            <1> sysvideo_15_96:
                            <1>         ; bl = 7, 8, 9, 10, 11, 12, 13
                            <1>         ; (bl > 13 is invalid for current kernel version)
                            <1>
                            <1>         ;cmp   bl, 13
                            <1>         ;ja    short sysvideo_15_91
                            <1>
                            <1>         ; 01/01/2021
0000ECEE 80FB0E             <1>         cmp    bl, 14
0000ECF1 0F8372050000       <1>         jnb    sysvideo_15_164
                            <1>
                            <1>         ; eax = LFB video page size (w*h*4) in bytes
                            <1>
0000ECF7 5E                 <1>         pop    esi ; * ; LFB base/start address
0000ECF8 91                 <1>         xchg   eax, ecx ; ecx = byte count
```

```
                                <1>                     ; eax = dword for and/or/xor
0000ECF9 890D[64030300]         <1>       mov   [u.r0], ecx ; return with
                                <1>                       ; byte count of display page
                                <1>                       ; (eax > 0 -> OK)
0000ECFF C1E902                 <1>       shr   ecx, 2 ; dword count
                                <1>
0000ED02 80FB0A                 <1>       cmp   bl, 10
0000ED05 7223                   <1>       jb    short sysvideo_15_100 ; 7, 8, 9
0000ED07 740C                   <1>       je    short sysvideo_15_97 ; 10
0000ED09 80FB0C                 <1>       cmp   bl, 12
0000ED0C 720E                   <1>       jb    short sysvideo_15_98 ; 11
0000ED0E 7413                   <1>       je    short sysvideo_15_99 ; 12
                                <1>
                                <1>       ; BL = 13 = NOT display page dwords
                                <1>
0000ED10 E924FEFFFF             <1>       jmp   sysvideo_15_79
                                <1>
                                <1>       ; BL = 10 = INC display page dwords
                                <1> sysvideo_15_97:
0000ED15 FF06                   <1>       inc   dword [esi]
0000ED17 AD                     <1>       lodsd
0000ED18 E2FB                   <1>       loop  sysvideo_15_97
0000ED1A EB2A                   <1>       jmp   short sysvideo_15_104
                                <1>
                                <1>       ; BL = 11 = DEC display page dwords
                                <1> sysvideo_15_98:
0000ED1C FF0E                   <1>       dec   dword [esi]
0000ED1E AD                     <1>       lodsd
0000ED1F E2FB                   <1>       loop  sysvideo_15_98
0000ED21 EB23                   <1>       jmp   short sysvideo_15_104
                                <1>
                                <1>       ; BL = 12 = NEG display page dwords
                                <1> sysvideo_15_99:
0000ED23 F71E                   <1>       neg   dword [esi]
0000ED25 AD                     <1>       lodsd
0000ED26 E2FB                   <1>       loop  sysvideo_15_99
0000ED28 EB1C                   <1>       jmp   short sysvideo_15_104
                                <1>
                                <1> sysvideo_15_100:
                                <1>       ; bl = 7, 8, 9
                                <1>       ; ecx = pixel count
0000ED2A 89C2                   <1>       mov   edx, eax
                                <1>             ; edx = dword for and/or/xor
                                <1>
0000ED2C 80FB08                 <1>       cmp   bl, 8
0000ED2F 7409                   <1>       je    short sysvideo_15_102
0000ED31 770E                   <1>       ja    short sysvideo_15_103
                                <1>
                                <1>       ; BL = 7 = AND display page dwords with ECX
                                <1> sysvideo_15_101:
0000ED33 2116                   <1>       and   [esi], edx
0000ED35 AD                     <1>       lodsd
0000ED36 E2FB                   <1>       loop  sysvideo_15_101
                                <1>       ;jmp  sysret
0000ED38 EB0C                   <1>       jmp   short sysvideo_15_104
                                <1>
                                <1>       ; BL = 8 = OR display page dwords with ECX
                                <1> sysvideo_15_102:
0000ED3A 0916                   <1>       or    [esi], edx
0000ED3C AD                     <1>       lodsd
0000ED3D E2FB                   <1>       loop  sysvideo_15_102
                                <1>       ;jmp  sysret
0000ED3F EB05                   <1>       jmp   short sysvideo_15_104
                                <1>
                                <1>       ; BL = 9 = XOR display page dwords with ECX
                                <1> sysvideo_15_103:
0000ED41 3116                   <1>       xor   [esi], edx
0000ED43 AD                     <1>       lodsd
0000ED44 E2FB                   <1>       loop  sysvideo_15_103
                                <1> sysvideo_15_104:
0000ED46 E9A1EBFFFF             <1>       jmp   sysret
                                <1>
                                <1> sysvideo_15_105:
                                <1>       ; bh = bits per pixel (bpp) = 24
                                <1>
                                <1>       ; eax = linear frame buffer (VIDEO PAGE) pixel count
                                <1>       ;     (1/3 of video page size in bytes)
                                <1>
0000ED4B 52                     <1>       push  edx
0000ED4C 89C2                   <1>       mov   edx, eax
0000ED4E D1E2                   <1>       shl   edx, 1 ; * 2
0000ED50 01D0                   <1>       add   eax, edx ; * 3 (24 bits per pixel)
0000ED52 5A                     <1>       pop   edx
                                <1>
                                <1>       ; eax = LFB video page byte count (3 * pixel count)
                                <1>
                                <1>       ; sub function for sysvideo function bh = 2
0000ED53 20DB                   <1>       and   bl, bl ; 0 ?
0000ED55 751B                   <1>       jnz   short sysvideo_15_106 ; no
                                <1>
                                <1>       ; BL = 0 = Fill color (color in ECX)
                                <1>
0000ED57 5F                     <1>       pop   edi ; * ; LFB address
                                <1>
0000ED58 A3[64030300]           <1>       mov   [u.r0], eax ; return value to user
                                <1>
0000ED5D 91                     <1>       xchg  eax, ecx ; eax = color, ecx = pixel count
                                <1>
                                <1>       ; clear bits 24 to 31 even if they are not zero
                                <1>       ;and  eax, 0FFFFFFh
                                <1>
                                <1>       ; 24 bit pixels
                                <1>
                                <1>       ; ecx = byte count
                                <1>
```

```
5424 0000ED5E 89C3              <1>       mov    ebx, eax
5425 0000ED60 C1EB10            <1>       shr    ebx, 16
5426                            <1>
5427                            <1> sysvideo_15_105_0:
5428 0000ED63 66AB              <1>       stosw  ; ax
5429 0000ED65 881F              <1>       mov    [edi], bl ; bit 16 to 23 of eax
5430 0000ED67 47                <1>       inc    edi
5431 0000ED68 83E903            <1>       sub    ecx, 3
5432                            <1>       ;jnz   short sysvideo_15_105_0
5433 0000ED6B 77F6              <1>       ja     short sysvideo_15_105_0
5434                            <1>
5435                            <1>       ; [u.r0] = transfer (fill) count in bytes
5436 0000ED6D E97AEBFFFF        <1>       jmp    sysret
5437                            <1>
5438                            <1> sysvideo_15_106:
5439                            <1>       ; 27/12/2020
5440                            <1>
5441                            <1>       ; eax = linear frame buffer (VIDEO PAGE) byte count
5442                            <1>       ;       (3 * pixel count)
5443                            <1>
5444 0000ED72 80FB02            <1>       cmp    bl, 2
5445 0000ED75 7725              <1>       ja     short sysvideo_15_111
5446 0000ED77 7217              <1>       jb     short sysvideo_15_109
5447                            <1>
5448                            <1>       ; BL = 2 = system to user video/display page transfer
5449 0000ED79 5E                <1>       pop    esi ; * ; LFB address
5450 0000ED7A 89CF              <1>       mov    edi, ecx ; user's buffer
5451                            <1> sysvideo_15_107:
5452 0000ED7C 89C1              <1>       mov    ecx, eax ; LFB video page byte count
5453 0000ED7E E841250000        <1>       call   transfer_to_user_buffer ; fast transfer
5454                            <1> sysvideo_15_108:
5455                            <1>       ;jc    sysret ; [u.r0] = 0
5456 0000ED83 72C1              <1>       jc     short sysvideo_15_104
5457 0000ED85 890D[64030300]    <1>       mov    [u.r0], ecx ; actual transfer count
5458 0000ED8B E95CEBFFFF        <1>       jmp    sysret
5459                            <1>
5460                            <1> sysvideo_15_109:
5461                            <1>       ; BL = 1 = user to system video/display page transfer
5462 0000ED90 5F                <1>       pop    edi ; * ; LFB address
5463 0000ED91 89CE              <1>       mov    esi, ecx ; user's buffer
5464                            <1> sysvideo_15_110:
5465 0000ED93 89C1              <1>       mov    ecx, eax ; LFB video page byte count
5466 0000ED95 E874250000        <1>       call   transfer_from_user_buffer ; fast transfer
5467 0000ED9A EBE7              <1>       jmp    short sysvideo_15_108
5468                            <1>
5469                            <1> sysvideo_15_111:
5470 0000ED9C 80FB03            <1>       cmp    bl, 3
5471 0000ED9F 7777              <1>       ja     short sysvideo_15_117
5472                            <1>
5473                            <1>       ; BL = 3 = NOT bits in window (ECX, EDX)
5474 0000EDA1 5E                <1>       pop    esi ; * ; LFB address
5475                            <1>
5476 0000EDA2 39CA              <1>       cmp    edx, ecx ; bottom-right > top-left ?
5477 0000EDA4 7315              <1>       jnb    short sysvideo_15_113 ; window
5478                            <1>
5479 0000EDA6 09D2              <1>       or     edx, edx
5480 0000EDA8 7569              <1>       jnz    short sysvideo_15_116  ; invalid !
5481                            <1>
5482                            <1>       ; full screen (update) ; edx = 0, ecx > 0
5483 0000EDAA 89C1              <1>       mov    ecx, eax ; LFB video page byte count
5484 0000EDAC A3[64030300]      <1>       mov    [u.r0], eax
5485                            <1> sysvideo_15_112:
5486 0000EDB1 F616              <1>       not    byte [esi]
5487 0000EDB3 46                <1>       inc    esi
5488 0000EDB4 E2FB              <1>       loop   sysvideo_15_112
5489 0000EDB6 E931EBFFFF        <1>       jmp    sysret
5490                            <1>
5491                            <1> sysvideo_15_113:
5492                            <1>       ; 29/12/2020
5493                            <1>       ; fix dx to bp-1 if dx >= bp
5494 0000EDBB E8690D0000        <1>       call   sysvideo_15_200
5495                            <1>       ;
5496                            <1>       ; 29/12/2020
5497 0000EDC0 89F7              <1>       mov    edi, esi
5498 0000EDC2 01C7              <1>       add    edi, eax ; end 3 bytes of video page + 3
5499 0000EDC4 0FB7C2            <1>       movzx  eax, dx ; bottom right column
5500 0000EDC7 6629C8            <1>       sub    ax, cx  ; - top left column
5501 0000EDCA 7247              <1>         jb      short sysvideo_15_116 ; invalid
5502 0000EDCC 40                <1>       inc    eax ; same column no == 1 column
5503 0000EDCD 50                <1>       push   eax ; pixel count per window row
5504 0000EDCE 52                <1>       push   edx
5505                            <1>       ;movzx ebx, word [ebp+2] ; screen width
5506                            <1>       ; 30/12/2020
5507                            <1>       ; calculate start offset
5508 0000EDCF E8600D0000        <1>       call   sysvideo_15_202
5509                            <1>              ; eax = pixel position
5510 0000EDD4 89C2              <1>       mov    edx, eax
5511 0000EDD6 D1E0              <1>       shl    eax, 1 ; * 2
5512 0000EDD8 01D0              <1>       add    eax, edx ; * 3
5513                            <1>              ; eax = byte position
5514 0000EDDA 01C6              <1>       add    esi, eax ; start address
5515 0000EDDC 59                <1>       pop    ecx ; edx
5516                            <1>       ; 30/12/2020
5517                            <1>       ; calculate start offset
5518 0000EDDD E8540D0000        <1>       call   sysvideo_15_203
5519                            <1>              ; eax = pixel position
5520 0000EDE2 89C2              <1>       mov    edx, eax
5521 0000EDE4 D1E0              <1>       shl    eax, 1 ; * 2
5522 0000EDE6 01D0              <1>       add    eax, edx ; * 3
5523                            <1>              ; eax = byte position
5524 0000EDE8 5A                <1>       pop    edx ; pixel count per window row
5525                            <1>       ; eax = offset of the last 3 bytes to be converted
5526 0000EDE9 01F0              <1>       add    eax, esi
5527 0000EDEB 39C7              <1>       cmp    edi, eax
5528 0000EDED 7624              <1>       jna    short sysvideo_15_116 ; out of video page
```

```
5529 0000EDEF 89C7              <1>       mov    edi, eax ; stop address
5530 0000EDF1 29D3              <1>       sub    ebx, edx ; screen width - window width
5531 0000EDF3 89D8              <1>       mov    eax, ebx
5532 0000EDF5 01DB              <1>       add    ebx, ebx ; * 2
5533 0000EDF7 01C3              <1>       add    ebx, eax ; * 3
5534                            <1>              ; ebx = screen-window difference in bytes
5535 0000EDF9 89D0              <1>       mov    eax, edx
5536 0000EDFB 01D2              <1>       add    edx, edx ; * 2
5537 0000EDFD 01C2              <1>       add    edx, eax ; * 3
5538                            <1>              ; edx = row size in bytes
5539 0000EDFF 4E                <1>       dec    esi
5540                            <1> sysvideo_15_114:
5541 0000EE00 89D1              <1>       mov    ecx, edx ; window width (byte count)
5542                            <1> sysvideo_15_115:
5543 0000EE02 46                <1>       inc    esi
5544 0000EE03 F616              <1>       not    byte [esi]
5545 0000EE05 E2FB              <1>       loop   sysvideo_15_115
5546 0000EE07 0115[64030300]    <1>       add    [u.r0], edx ; 1 pixel = 3 bytes
5547 0000EE0D 01DE              <1>       add    esi, ebx ; add difference for next row
5548                            <1>       ;
5549 0000EE0F 39FE              <1>       cmp    esi, edi ; stop address (included in loop)
5550 0000EE11 76ED              <1>       jna    short sysvideo_15_114
5551                            <1> sysvideo_15_116:
5552 0000EE13 E9D4EAFFFF        <1>       jmp    sysret
5553                            <1>
5554                            <1> sysvideo_15_117:
5555 0000EE18 80FB04            <1>       cmp    bl, 4
5556 0000EE1B 0F878C000000      <1>         ja   sysvideo_15_121
5557                            <1>
5558                            <1>       ; BL = 4 = window copy (system to system)
5559                            <1>
5560                            <1>       ; eax = LFB video page byte count
5561                            <1>       ;       (3 * pixel count)
5562                            <1>
5563 0000EE21 5B                <1>       pop    ebx ; * ; LFB address
5564                            <1>
5565 0000EE22 39DE              <1>       cmp    esi, ebx
5566 0000EE24 72ED              <1>       jb     short sysvideo_15_116
5567                            <1>
5568 0000EE26 39DF              <1>       cmp    edi, ebx
5569 0000EE28 72E9              <1>       jb     short sysvideo_15_116
5570 0000EE2A 01C3              <1>       add    ebx, eax ; **
5571 0000EE2C 39DE              <1>       cmp    esi, ebx
5572 0000EE2E 73E3              <1>       jnb    short sysvideo_15_116
5573 0000EE30 39DF              <1>       cmp    edi, ebx
5574 0000EE32 73DF              <1>       jnb    short sysvideo_15_116
5575                            <1>
5576 0000EE34 39CA              <1>       cmp    edx, ecx ; bottom-right > top-left ?
5577 0000EE36 7315              <1>       jnb    short sysvideo_15_118 ; window
5578                            <1>
5579 0000EE38 09D2              <1>       or     edx, edx
5580 0000EE3A 75D7              <1>       jnz    short sysvideo_15_116 ; invalid !
5581                            <1>
5582 0000EE3C A3[64030300]      <1>       mov    [u.r0], eax
5583 0000EE41 89C1              <1>       mov    ecx, eax ; byte count
5584 0000EE43 C1E902            <1>       shr    ecx, 2
5585                            <1>       ; full screen copy ; edx = 0, ecx > 0
5586 0000EE46 F3A5              <1>       rep    movsd
5587 0000EE48 E99FEAFFFF        <1>       jmp    sysret
5588                            <1> sysvideo_15_118:
5589                            <1>       ; 29/12/2020
5590                            <1>       ; fix dx to bp-1 if dx >= bp
5591 0000EE4D E8D70C0000        <1>       call   sysvideo_15_200
5592                            <1>       ;
5593                            <1>       ; 29/12/2020
5594 0000EE52 6639CA            <1>       cmp    dx, cx
5595 0000EE55 72BC              <1>         jb      short sysvideo_15_116 ; invalid
5596                            <1>
5597 0000EE57 53                <1>       push   ebx ; end 3 bytes of video page + 3
5598 0000EE58 29C3              <1>       sub    ebx, eax ; **
5599 0000EE5A 53                <1>       push   ebx ; LFB start/base address
5600                            <1>
5601 0000EE5B 0FB7C2            <1>       movzx  eax, dx ; bottom right column
5602 0000EE5E 6629C8            <1>       sub    ax, cx  ; - top left column
5603 0000EE61 40                <1>       inc    eax ; same column no == 1 column
5604 0000EE62 50                <1>       push   eax ; pixel count per window row
5605 0000EE63 52                <1>       push   edx
5606                            <1>       ;movzx ebx, word [ebp+2] ; screen width
5607                            <1>       ; 30/12/2020
5608                            <1>       ; calculate start offset
5609 0000EE64 E8CB0C0000        <1>       call   sysvideo_15_202
5610                            <1>              ; eax = pixel position
5611 0000EE69 89C2              <1>       mov    edx, eax
5612 0000EE6B D1E0              <1>       shl    eax, 1 ; * 2
5613 0000EE6D 01D0              <1>       add    eax, edx ; * 3
5614                            <1>              ; eax = byte position
5615 0000EE6F 01C7              <1>       add    edi, eax ; start address
5616 0000EE71 01C6              <1>       add    esi, eax
5617 0000EE73 59                <1>       pop    ecx ; edx
5618                            <1>       ; 30/12/2020
5619                            <1>       ; calculate start offset
5620 0000EE74 E8BD0C0000        <1>       call   sysvideo_15_203
5621                            <1>              ; eax = pixel position
5622 0000EE79 89C2              <1>       mov    edx, eax
5623 0000EE7B D1E0              <1>       shl    eax, 1 ; * 2
5624 0000EE7D 01D0              <1>       add    eax, edx ; * 3
5625                            <1>              ; eax = byte position
5626 0000EE7F 5A                <1>       pop    edx ; pixel count per window row
5627 0000EE80 59                <1>       pop    ecx ; LFB start/base address
5628 0000EE81 01C8              <1>       add    eax, ecx ; stop address
5629 0000EE83 59                <1>       pop    ecx ; the last 3 bytes of LFB + 3
5630 0000EE84 39C8              <1>       cmp    eax, ecx
5631 0000EE86 738B              <1>       jnb    short sysvideo_15_116
5632                            <1>
5633                            <1>       ;push  edi ; start address
```

```
5634                                <1>       ;push  eax ; stop address (included)
5635 0000EE88 29D3                 <1>       sub    ebx, edx ; screen width - window width
5636 0000EE8A 89D9                 <1>       mov    ecx, ebx
5637 0000EE8C 01DB                 <1>       add    ebx, ebx ; * 2
5638 0000EE8E 01CB                 <1>       add    ebx, ecx ; * 3
5639                                <1>              ; ebx = screen-window difference in bytes
5640 0000EE90 89D1                 <1>       mov    ecx, edx
5641 0000EE92 01D2                 <1>       add    edx, edx ; * 2
5642 0000EE94 01CA                 <1>       add    edx, ecx ; * 3
5643                                <1>              ; edx = row size in bytes
5644                                <1> sysvideo_15_119:
5645 0000EE96 89D1                 <1>       mov    ecx, edx
5646 0000EE98 F3A4                 <1>       rep    movsb
5647 0000EE9A 0115[64030300]       <1>       add    [u.r0], edx ; 1 pixel = 4 bytes
5648                                <1>       ;or    ebx, ebx
5649                                <1>       ;jz    short sysvideo_15_120
5650 0000EEA0 01DF                 <1>       add    edi, ebx ; add difference (for next row)
5651 0000EEA2 01DE                 <1>       add    esi, ebx
5652                                <1> ;sysvideo_15_120:
5653                                <1>       ;cmp   edi, [esp] ; stop addr (included in loop)
5654 0000EEA4 39C7                 <1>       cmp    edi, eax
5655 0000EEA6 76EE                 <1>       jna    short sysvideo_15_119
5656                                <1> sysvideo_15_120:
5657                                <1>       ;pop   ebx ; stop address
5658                                <1>       ;pop   edi ; start address
5659 0000EEA8 E93FEAFFFF           <1>       jmp    sysret
5660                                <1>
5661                                <1> sysvideo_15_121:
5662 0000EEAD 80FB05               <1>       cmp    bl, 5
5663 0000EEB0 776A                 <1>         ja      short sysvideo_15_126
5664                                <1>
5665                                <1>       ; BL = 5 = window copy (user to system)
5666                                <1>
5667                                <1>       ; eax = LFB video page byte count
5668                                <1>       ;      (3 * pixel count)
5669                                <1>
5670 0000EEB2 5F                   <1>       pop    edi ; * ; LFB address
5671                                <1>
5672                                <1>       ; esi = user buffer (in user's memory space)
5673 0000EEB3 39CA                 <1>       cmp    edx, ecx ; bottom-right > top-left ?
5674 0000EEB5 7309                 <1>              jnb    short sysvideo_15_122 ; window
5675                                <1>
5676 0000EEB7 09D2                 <1>       or     edx, edx
5677                                <1>       ;jnz   sysret  ; invalid !
5678 0000EEB9 75ED                 <1>       jnz    short sysvideo_15_120
5679                                <1>
5680                                <1>       ; full screen copy ; edx = 0, ecx > 0
5681 0000EEBB E9D3FEFFFF           <1>       jmp    sysvideo_15_110 ; full screen copy
5682                                <1>
5683                                <1> sysvideo_15_122:
5684                                <1>       ; 29/12/2020
5685                                <1>       ; fix dx to bp-1 if dx >= bp
5686 0000EEC0 E8640C0000           <1>       call   sysvideo_15_200
5687                                <1>       ;
5688                                <1>       ; 29/12/2020
5689 0000EEC5 6639CA               <1>       cmp    dx, cx
5690 0000EEC8 72DE                 <1>       jb     short sysvideo_15_120 ; invalid !
5691                                <1>
5692                                <1>       ; 01/01/2021
5693 0000EECA 01F8                 <1>       add    eax, edi
5694 0000EECC 50                   <1>       push   eax ; the last 3 bytes of LFB + 3
5695 0000EECD 57                   <1>       push   edi ; LFB start/base address
5696                                <1>
5697 0000EECE 0FB7C2               <1>       movzx  eax, dx ; bottom right column
5698 0000EED1 6629C8               <1>       sub    ax, cx  ; - top left column
5699 0000EED4 40                   <1>       inc    eax ; same column no == 1 column
5700 0000EED5 50                   <1>       push   eax ; pixel count per window row
5701                                <1>
5702 0000EED6 52                   <1>       push   edx
5703                                <1>       ;movzx ebx, word [ebp+2] ; screen width
5704                                <1>       ; 30/12/2020
5705                                <1>       ; calculate start offset
5706 0000EED7 E8580C0000           <1>       call   sysvideo_15_202
5707                                <1>              ; eax = pixel position
5708 0000EEDC 89C2                 <1>       mov    edx, eax
5709 0000EEDE D1E0                 <1>       shl    eax, 1 ; * 2
5710 0000EEE0 01D0                 <1>       add    eax, edx ; * 3
5711                                <1>              ; eax = byte position
5712 0000EEE2 01C7                 <1>       add    edi, eax ; start address
5713 0000EEE4 59                   <1>       pop    ecx ; edx
5714                                <1>       ; 30/12/2020
5715                                <1>       ; calculate end offset
5716 0000EEE5 E84C0C0000           <1>       call   sysvideo_15_203
5717                                <1>              ; eax = pixel position
5718 0000EEEA 89C2                 <1>       mov    edx, eax
5719 0000EEEC D1E0                 <1>       shl    eax, 1 ; * 2
5720 0000EEEE 01D0                 <1>       add    eax, edx ; * 3
5721                                <1>              ; eax = byte position
5722 0000EEF0 5A                   <1>       pop    edx ; pixel count per window row
5723 0000EEF1 59                   <1>       pop    ecx ; LFB start/base address
5724 0000EEF2 01C8                 <1>       add    eax, ecx ; stop address
5725 0000EEF4 59                   <1>       pop    ecx ; the last 3 bytes of LFB + 3
5726 0000EEF5 39C8                 <1>       cmp    eax, ecx
5727                                <1>       ;jnb   sysret
5728 0000EEF7 73AF                 <1>       jnb    short sysvideo_15_120
5729                                <1>
5730                                <1>       ; 29/12/2020
5731 0000EEF9 89D9                 <1>       mov    ecx, ebx
5732 0000EEFB 01DB                 <1>       add    ebx, ebx ; * 2
5733 0000EEFD 01CB                 <1>       add    ebx, ecx ; * 3
5734                                <1>              ; ebx = screen-window difference in bytes
5735 0000EEFF 89D1                 <1>       mov    ecx, edx
5736 0000EF01 01D2                 <1>       add    edx, edx ; * 2
5737 0000EF03 01CA                 <1>       add    edx, ecx ; * 3
5738                                <1>              ; edx = row size in bytes
```

```
                        <1>       ;push  edi ; start address
0000EF05 50             <1>       push   eax ; stop address (included)
                        <1> sysvideo_15_123:
0000EF06 89D1           <1>       mov    ecx, edx ; byte count
                        <1>       ; user to system video/display page window transfer
                        <1>       ; esi =      user buffer
0000EF08 E801240000     <1>       call   transfer_from_user_buffer ; fast transfer
0000EF0D 7207           <1>       jc     short sysvideo_15_124
                        <1>              ; ecx = actual transfer count (= ecx input)
                        <1>       ;add   [u.r0], ecx
                        <1>       ;add   edi, ebx ; next row in video memory window
                        <1>       ;add   esi, ecx ; next window row in user's buffer
                        <1>       ;cmp   edi, [esp] ; stop addr (included in loop)
                        <1>       ; 30/12/2020
0000EF0F E82F0C0000     <1>       call   sysvideo_15_204
0000EF14 76F0           <1>       jna    short sysvideo_15_123
                        <1> sysvideo_15_124:
0000EF16 5B             <1>       pop    ebx ; stop address
                        <1>       ;pop   edi ; start address
                        <1> sysvideo_15_125:
0000EF17 E9D0E9FFFF     <1>       jmp    sysret
                        <1>
                        <1> sysvideo_15_126:
0000EF1C 80FB06         <1>       cmp    bl, 6
0000EF1F 776A           <1>       ja     short sysvideo_15_129
                        <1>
                        <1>       ; BL = 6 = window copy (system to user)
                        <1>
                        <1>       ; eax = LFB video page byte count
                        <1>       ;      (3 * pixel count)
                        <1>
0000EF21 89F7           <1>       mov    edi, esi ; user buffer
                        <1>
0000EF23 5E             <1>       pop    esi ; * ; LFB address
                        <1>
                        <1>       ; edi = user buffer (in user's memory space)
0000EF24 39CA           <1>       cmp    edx, ecx ; bottom-right > top-left ?
0000EF26 7309           <1>         jnb     short sysvideo_15_127
                        <1>
0000EF28 21D2           <1>       and    edx, edx
                        <1>       ;jnz   sysret ; invalid !
0000EF2A 75EB           <1>       jnz    short sysvideo_15_125
                        <1>
                        <1>       ; full screen copy  ; edx = 0, ecx > 0
0000EF2C E94BFEFFFF     <1>       jmp    sysvideo_15_107
                        <1>
                        <1> sysvideo_15_127:
                        <1>       ; 29/12/2020
                        <1>       ; fix dx to bp-1 if dx >= bp
0000EF31 E8F30B0000     <1>       call   sysvideo_15_200
                        <1>       ;
                        <1>       ; 29/12/2020
0000EF36 6639CA         <1>       cmp    dx, cx
0000EF39 72DC           <1>       jb     short sysvideo_15_125 ; invalid !
                        <1>
0000EF3B 89F3           <1>       mov    ebx, esi ; LFB address
0000EF3D 01D8           <1>       add    eax, ebx
0000EF3F 50             <1>       push   eax ; the last 3 bytes of LFB + 3
0000EF40 53             <1>       push   ebx ; LFB start/base address
                        <1>
0000EF41 0FB7C2         <1>       movzx  eax, dx ; bottom right column
0000EF44 6629C8         <1>       sub    ax, cx  ; - top left column
0000EF47 40             <1>       inc    eax ; same column no == 1 column
0000EF48 50             <1>       push   eax ; pixel count per window row
                        <1>
0000EF49 52             <1>       push   edx
                        <1>       ;movzx ebx, word [ebp+2] ; screen width
                        <1>       ; 30/12/2020
                        <1>       ; calculate start offset
0000EF4A E8E50B0000     <1>       call   sysvideo_15_202
                        <1>              ; eax = pixel position
0000EF4F 89C2           <1>       mov    edx, eax
0000EF51 D1E0           <1>       shl    eax, 1 ; * 2
0000EF53 01D0           <1>       add    eax, edx ; * 3
                        <1>              ; eax = byte position
0000EF55 01C6           <1>       add    esi, eax ; start address
0000EF57 59             <1>       pop    ecx ; edx
                        <1>       ; 30/12/2020
                        <1>       ; calculate end offset
0000EF58 E8D90B0000     <1>       call   sysvideo_15_203
                        <1>              ; eax = pixel position
0000EF5D 89C2           <1>       mov    edx, eax
0000EF5F D1E0           <1>       shl    eax, 1 ; * 2
0000EF61 01D0           <1>       add    eax, edx ; * 3
                        <1>              ; eax = byte position
0000EF63 5A             <1>       pop    edx ; pixel count per window row
0000EF64 59             <1>       pop    ecx ; LFB start/base address
0000EF65 01C8           <1>       add    eax, ecx ; stop address
0000EF67 59             <1>       pop    ecx ; the last 3 bytes of LFB + 3
0000EF68 39C8           <1>       cmp    eax, ecx
                        <1>       ;jnb   sysret
0000EF6A 73AB           <1>       jnb    short sysvideo_15_125
                        <1>
                        <1>       ; 29/12/2020
                        <1>       ;push  esi ; start address
0000EF6C 50             <1>       push   eax ; stop address (included)
                        <1>
0000EF6D 89D8           <1>       mov    eax, ebx
0000EF6F 01DB           <1>       add    ebx, ebx ; * 2
0000EF71 01C3           <1>       add    ebx, eax ; * 3
                        <1>              ; ebx = screen-window difference in bytes
0000EF73 89D0           <1>       mov    eax, edx
0000EF75 01D2           <1>       add    edx, edx ; * 2
0000EF77 01C2           <1>       add    edx, eax ; * 3
                        <1>              ; edx = row size in bytes
```

```
                                <1> sysvideo_15_128:
0000EF79 89D1                   <1>         mov     ecx, edx ; byte count
                                <1>         ; user to system video/display page window transfer
                                <1>         ; esi =       user buffer
0000EF7B E844230000             <1>         call    transfer_to_user_buffer ; fast transfer
                                <1>         ;jc     short sysvideo_15_128_0
0000EF80 7294                   <1>         jc      short sysvideo_15_124
                                <1>         ;add    [u.r0], ecx
                                <1>         ;add    edi, ebx ; next row in video memory window
                                <1>         ;add    esi, ecx ; next window row in user's buffer
                                <1>         ;cmp    edi, [esp] ; stop addr (included in loop)
                                <1>         ; 30/12/2020
0000EF82 E8BC0B0000             <1>         call    sysvideo_15_204
0000EF87 76F0                   <1>         jna     short sysvideo_15_128
0000EF89 EB8B                   <1>         jmp     sysvideo_15_124
                                <1>
                                <1> sysvideo_15_129:
                                <1>         ; bl = 7, 8, 9, 10, 11, 12, 13
                                <1>         ; (bl > 13 is invalid for current kernel version)
                                <1>
                                <1>         ;cmp    bl, 13
                                <1>         ;ja     short sysvideo_15_124
                                <1>
                                <1>         ; 01/01/2021
0000EF8B 80FB0E                 <1>         cmp     bl, 14
0000EF8E 0F83F8030000           <1>         jnb     sysvideo_15_177
                                <1>
                                <1>         ; eax = LFB video page size (w*h*3) in bytes
                                <1>
0000EF94 5E                     <1>         pop     esi ; * ; LFB base/start address
                                <1>
0000EF95 91                     <1>         xchg    eax, ecx ; ecx = byte count
                                <1>                          ; eax = 3 bytes for and/or/xor
0000EF96 890D[64030300]         <1>         mov     [u.r0], ecx ; return with
                                <1>                             ; byte count of display page
                                <1>                             ; (eax > 0 -> OK)
0000EF9C 80FB0A                 <1>         cmp     bl, 10
0000EF9F 7249                   <1>         jb      short sysvideo_15_133 ; 7, 8, 9
0000EFA1 740C                   <1>         je      short sysvideo_15_130 ; 10
0000EFA3 80FB0C                 <1>         cmp     bl, 12
0000EFA6 7219                   <1>         jb      short sysvideo_15_131 ; 11
0000EFA8 7429                   <1>         je      short sysvideo_15_132 ; 12
                                <1>
                                <1>         ; BL = 13 = NOT display page dwords
                                <1>
0000EFAA E902FEFFFF             <1>         jmp     sysvideo_15_112
                                <1>
                                <1>         ; BL = 10 = INC display page dwords
                                <1> sysvideo_15_130:
0000EFAF FE06                   <1>         inc     byte [esi]
0000EFB1 7504                   <1>         jnz     short sysvideo_15_130_0
0000EFB3 66FF4602               <1>         inc     word [esi+2]
                                <1> sysvideo_15_130_0:
0000EFB7 83C603                 <1>         add     esi, 3
0000EFBA 83E903                 <1>         sub     ecx, 3
0000EFBD 77F0                   <1>         ja      short sysvideo_15_130
0000EFBF EB6D                   <1>         jmp     short sysvideo_15_137
                                <1>
                                <1>         ; BL = 11 = DEC display page dwords
                                <1> sysvideo_15_131:
0000EFC1 FE0E                   <1>         dec     byte [esi]
0000EFC3 7504                   <1>         jnz     short sysvideo_15_131_0
0000EFC5 66FF4E02               <1>         dec     word [esi+2]
                                <1> sysvideo_15_131_0:
0000EFC9 83C603                 <1>         add     esi, 3
0000EFCC 83E903                 <1>         sub     ecx, 3
0000EFCF 77F0                   <1>         ja      short sysvideo_15_131
0000EFD1 EB5B                   <1>         jmp     short sysvideo_15_137
                                <1>
                                <1>         ; BL = 12 = NEG display page dwords
                                <1> sysvideo_15_132:
                                <1>         ; 01/01/2021
                                <1>         ;mov    edx, 00FFFFFFh
                                <1> ;sysvideo_15_132_0:
0000EFD3 8B06                   <1>         mov     eax, [esi]
                                <1>         ;and    eax, edx
0000EFD5 F7D8                   <1>         neg     eax
0000EFD7 668906                 <1>         mov     [esi], ax
0000EFDA C1E810                 <1>         shr     eax, 16
0000EFDD 884602                 <1>         mov     [esi+2], al
0000EFE0 83C603                 <1>         add     esi, 3
0000EFE3 83E903                 <1>         sub     ecx, 3
                                <1>         ;ja     short sysvideo_15_132_0
0000EFE6 77EB                   <1>         ja      short sysvideo_15_132
0000EFE8 EB44                   <1>         jmp     short sysvideo_15_137
                                <1>
                                <1> sysvideo_15_133:
                                <1>         ; bl = 7, 8, 9
                                <1>         ; ecx = pixel count
                                <1>         ; 28/12/2020
0000EFEA 80FB08                 <1>         cmp     bl, 8
0000EFED 7417                   <1>         je      short sysvideo_15_135
0000EFEF 772A                   <1>         ja      short sysvideo_15_136
                                <1>
                                <1>         ; BL = 7 = AND display page dwords with ECX
                                <1> sysvideo_15_134:
                                <1>         ; 24 bit AND
0000EFF1 662106                 <1>         and     [esi], ax
0000EFF4 C1C810                 <1>         ror     eax, 16
0000EFF7 46                     <1>         inc     esi
0000EFF8 46                     <1>         inc     esi
0000EFF9 2006                   <1>         and     [esi], al
0000EFFB C1C010                 <1>         rol     eax, 16
0000EFFE 46                     <1>         inc     esi
0000EFFF 83E903                 <1>         sub     ecx, 3
```

```
5949 0000F002 77ED              <1>       ja     short sysvideo_15_134
5950                            <1>       ;jmp   sysret
5951 0000F004 EB28              <1>       jmp    short sysvideo_15_137
5952                            <1>
5953                            <1>       ; BL = 8 = OR display page dwords with ECX
5954                            <1> sysvideo_15_135:
5955                            <1>       ; 24 bit OR
5956 0000F006 660906            <1>       or     [esi], ax
5957 0000F009 C1C810            <1>       ror    eax, 16
5958 0000F00C 46                <1>       inc    esi
5959 0000F00D 46                <1>       inc    esi
5960 0000F00E 0806              <1>       or     [esi], al
5961 0000F010 C1C010            <1>       rol    eax, 16
5962 0000F013 46                <1>       inc    esi
5963 0000F014 83E903            <1>       sub    ecx, 3
5964 0000F017 77ED              <1>       ja     short sysvideo_15_135
5965                            <1>       ;jmp   sysret
5966 0000F019 EB13              <1>       jmp    short sysvideo_15_137
5967                            <1>
5968                            <1>       ; BL = 9 = XOR display page dwords with ECX
5969                            <1> sysvideo_15_136:
5970                            <1>       ; 24 bit XOR
5971 0000F01B 663106            <1>       xor    [esi], ax
5972 0000F01E C1C810            <1>       ror    eax, 16
5973 0000F021 46                <1>       inc    esi
5974 0000F022 46                <1>       inc    esi
5975 0000F023 3006              <1>       xor    [esi], al
5976 0000F025 C1C010            <1>       rol    eax, 16
5977 0000F028 46                <1>       inc    esi
5978 0000F029 83E903            <1>       sub    ecx, 3
5979 0000F02C 77ED              <1>       ja     short sysvideo_15_136
5980                            <1> sysvideo_15_137:
5981 0000F02E E9B9E8FFFF        <1>       jmp    sysret
5982                            <1>
5983                            <1> sysvideo_15_138:
5984                            <1>       ; 02/01/2020
5985                            <1>       ; bh = 8 (function: bh = 2 or bh = 1)
5986                            <1>       ; 01/01/2021
5987 0000F033 80FB0E            <1>       cmp    bl, 14
5988                            <1>       ;ja    sysvideo_15_190 ; bl = 15
5989 0000F036 0F85B9040000      <1>       jne    sysvideo_15_190 ; bl = 15
5990                            <1>
5991                            <1>       ; BL = 14 = masked write of pixels
5992                            <1>       ;          (8 bpp)
5993                            <1>
5994                            <1>       ; eax = LFB video page byte count
5995                            <1>       ;      (pixel count)
5996                            <1>
5997                            <1>       ; esi = user's mask buffer address
5998                            <1>       ; edi = mask color
5999                            <1>
6000                            <1>       ; ECX Low 16 bits = Top left column (X1 position)
6001                            <1>       ; ECX High 16 bits = Top row (Y1 position)
6002                            <1>       ; EDX Low 16 bits = Bottom right column (X2 position)
6003                            <1>       ; EDX High 16 bits = Bottom row (Y2 position)
6004                            <1>
6005                            <1>       ; EBX bit 16 to 18 -> mask operation/option
6006                            <1>       ;      0 = new color (in edi)
6007                            <1>       ;      1 = add color (in edi) -up to 0FFh-
6008                            <1>       ;      2 = average color (in edi) -(current+edi)/2-
6009                            <1>       ;      3 = sub color (in edi) -down to 0-
6010                            <1>       ;      4 = inc      color -up to 0FFh-
6011                            <1>       ;      5 = dec      color -down to 0-
6012                            <1>       ;      6 = not color
6013                            <1>       ;      7 = neg color
6014                            <1>
6015 0000F03C 893D[7E120300]    <1>       mov    [maskcolor], edi
6016                            <1>
6017 0000F042 C1EB10            <1>       shr    ebx, 16
6018 0000F045 6683E307          <1>       and    bx, 7 ; mask option
6019 0000F049 C0E302            <1>       shl    bl, 2 ; * 4
6020 0000F04C 8B9B[F5F00000]    <1>       mov    ebx, [ebx+mask_op_sub_tbl_8]
6021                            <1>
6022                            <1>       ; ebp = screen width
6023                            <1>       ; ebx = function address
6024                            <1>
6025 0000F052 5F                <1>       pop    edi ; * ; LFB address
6026                            <1>
6027                            <1>       ; esi = user buffer (in user's memory space)
6028                            <1>       ;      (contains mask bits, 8 * window size)
6029                            <1>
6030 0000F053 39CA              <1>       cmp    edx, ecx ; bottom-right > top-left ?
6031 0000F055 730F              <1>              jnb    short sysvideo_15_139 ; window
6032                            <1>
6033 0000F057 09D2              <1>       or     edx, edx
6034                            <1>       ;jnz   sysret  ; invalid !
6035 0000F059 7554              <1>       jnz    short sysvideo_15_142
6036                            <1>
6037                            <1>       ; full screen masked write ; edx = 0, ecx > 0
6038                            <1>       ; eax = LFB video page size
6039 0000F05B 89C2              <1>       mov    edx, eax
6040 0000F05D 4A                <1>       dec    edx ; last byte of LFB video page
6041 0000F05E 01FA              <1>       add    edx, edi
6042                            <1>       ; edx = stop address
6043 0000F060 89E9              <1>       mov    ecx, ebp ; screen width
6044 0000F062 87DD              <1>       xchg   ebx, ebp
6045                            <1>       ; ebx = screen width
6046                            <1>       ; ebp = function address
6047 0000F064 EB33              <1>       jmp    short sysvideo_15_140
6048                            <1>
6049                            <1> sysvideo_15_139:
6050                            <1>       ; masked window write
6051                            <1>       ;cmp   dx, bp ; end column < screen width ?
6052                            <1>       ;jb    short sysvideo_15_139_0
6053                            <1>       ; fix bottom right position of the window
```

```
6054                                    <1>       ;mov   dx, bp
6055                                    <1>       ;dec   dx
6056 0000F066 E8BE0A0000                <1>       call   sysvideo_15_200
6057                                    <1> ;sysvideo_15_139_0:
6058 0000F06B 6639CA                    <1>       cmp    dx, cx
6059 0000F06E 723F                      <1>       jb     short sysvideo_15_142
6060                                    <1>
6061 0000F070 53                        <1>       push   ebx ; (call) sub-function address
6062                                    <1>
6063 0000F071 01F8                      <1>       add    eax, edi
6064                                    <1>
6065 0000F073 50                        <1>       push   eax ; the last byte of LFB + 1
6066 0000F074 57                        <1>       push   edi ; LFB start/base address
6067                                    <1>
6068 0000F075 0FB7C2                    <1>       movzx  eax, dx ; bottom right column
6069 0000F078 6629C8                    <1>       sub    ax, cx  ; - top left column
6070 0000F07B 40                        <1>       inc    eax ; same column no == 1 column
6071                                    <1>
6072 0000F07C 50                        <1>       push   eax ; pixel count per window row
6073                                    <1>
6074 0000F07D 52                        <1>       push   edx
6075                                    <1>       ;;movzx      ebx, word [ebp+2] ; screen width
6076                                    <1>       ;mov   ebx, ebp ; screen width
6077                                    <1>       ;mov   eax, ecx
6078                                    <1>       ;shr   eax, 16      ; top row
6079                                    <1>       ;mul   ebx
6080                                    <1>       ;mov   dx, cx ; top left column
6081                                    <1>       ;add   eax, edx
6082 0000F07E E8B10A0000                <1>       call   sysvideo_15_202
6083 0000F083 01C7                      <1>       add    edi, eax ; start address
6084 0000F085 59                        <1>       pop    ecx ; edx
6085                                    <1>
6086                                    <1>       ;mov   eax, ecx
6087                                    <1>       ;shr   eax, 16 ; bottom row
6088                                    <1>       ;mul   ebx
6089                                    <1>       ;mov   dx, cx ; bottom right column
6090                                    <1>       ;add   eax, edx
6091                                    <1>       ; 30/12/2020
6092                                    <1>       ; calculate end offset
6093 0000F086 E8AB0A0000                <1>       call   sysvideo_15_203
6094                                    <1>
6095 0000F08B 5A                        <1>       pop    edx ; pixel count per window row
6096 0000F08C 59                        <1>       pop    ecx ; LFB start/base address
6097                                    <1>
6098 0000F08D 01C8                      <1>       add    eax, ecx ; stop address
6099 0000F08F 59                        <1>       pop    ecx ; the last byte of LFB + 1
6100                                    <1>
6101 0000F090 5D                        <1>       pop    ebp ; (call) sub-function address
6102                                    <1>
6103 0000F091 39C8                      <1>       cmp    eax, ecx
6104 0000F093 7357                      <1>       jnb    short sysvideo_15_150 ; wrong pos !
6105                                    <1>
6106                                    <1>       ;push  edi ; start address
6107                                    <1>
6108 0000F095 89D1                      <1>       mov    ecx, edx ; byte count
6109 0000F097 89C2                      <1>       mov    edx, eax ; stop address
6110                                    <1>                       ; (the last byte included)
6111                                    <1> sysvideo_15_140:
6112                                    <1>       ; ebp = (masked) color change function address
6113                                    <1>       ; esi = user's buffer address (virtual)
6114                                    <1>       ; edi = window start address in LFB
6115                                    <1>       ; ebx = screen width
6116                                    <1>       ; edx = stop address
6117                                    <1>       ; ecx = window width in bytes
6118 0000F099 83C107                    <1>       add    ecx, 7
6119 0000F09C C1E903                    <1>       shr    ecx, 3 ; mask byte count (for user's buffer)
6120                                    <1> sysvideo_15_141:
6121                                    <1>       ; edx = stop address (the last byte included!)
6122 0000F09F 51                        <1>       push   ecx ; **
6123 0000F0A0 57                        <1>       push   edi ; * ; start address in LFB
6124                                    <1>       ; ecx = byte count (per row)
6125 0000F0A1 BF00600900                <1>       mov    edi, VBE3STACKADDR ; destination
6126                                    <1>       ; user to system video/display page window transfer
6127                                    <1>       ; esi =      user buffer
6128                                    <1>       ; edi = system buffer
6129 0000F0A6 E863220000                <1>       call   transfer_from_user_buffer ; fast transfer
6130                                    <1>              ; eax & ecx modified, other regs preserved
6131 0000F0AB 5F                        <1>       pop    edi ; * ; start address in LFB
6132 0000F0AC 7306                      <1>       jnc    short sysvideo_15_143
6133                                    <1>       ; memory error !
6134 0000F0AE 59                        <1>       pop    ecx ; **
6135                                    <1> sysvideo_15_142:
6136 0000F0AF E938E8FFFF                <1>       jmp    sysret
6137                                    <1> sysvideo_15_143:
6138 0000F0B4 56                        <1>       push   esi ; *** ; user's buffer address
6139 0000F0B5 BE00600900                <1>       mov    esi, VBE3STACKADDR ; source (in sys mem)
6140                                    <1> sysvideo_15_144:
6141 0000F0BA 53                        <1>       push   ebx ; **** ; screen width
6142 0000F0BB 57                        <1>       push   edi ; ***** ; window start address in LFB
6143 0000F0BC 8A1D[7E120300]            <1>       mov    bl, [maskcolor]
6144                                    <1> sysvideo_15_145:
6145 0000F0C2 B408                      <1>       mov    ah, 8 ; 8 mask bits for 8 pixels (8 bytes)
6146 0000F0C4 AC                        <1>       lodsb
6147                                    <1> sysvideo_15_146:
6148 0000F0C5 D0E8                      <1>       shr    al, 1
6149 0000F0C7 7402                      <1>       jz     short sysvideo_15_147
6150                                    <1>       ;mov   [edi], bl  ; color
6151 0000F0C9 FFD5                      <1>       call   ebp ; sub function (color change function)
6152                                    <1> sysvideo_15_147:
6153 0000F0CB FF05[64030300]            <1>       inc    dword [u.r0]
6154 0000F0D1 47                        <1>       inc    edi
6155 0000F0D2 39D7                      <1>       cmp    edi, edx ; stop address ?
6156 0000F0D4 7716                      <1>       ja     short sysvideo_15_150
6157 0000F0D6 FECC                      <1>       dec    ah
6158 0000F0D8 7405                      <1>       jz     short sysvideo_15_148
```

```
6159 0000F0DA 49                <1>       dec    ecx
6160 0000F0DB 75E8              <1>       jnz    short sysvideo_15_146
6161 0000F0DD EB03              <1>       jmp    short sysvideo_15_149
6162                            <1> sysvideo_15_148:
6163 0000F0DF 49                <1>       dec    ecx
6164 0000F0E0 75E0              <1>       jnz    short sysvideo_15_145
6165                            <1> sysvideo_15_149:
6166 0000F0E2 5F                <1>       pop    edi ; ***** ; cur row in video memory window
6167 0000F0E3 5B                <1>       pop    ebx ; **** ; screen width
6168 0000F0E4 5E                <1>       pop    esi ; ***  ; user's buffer address
6169 0000F0E5 59                <1>       pop    ecx ; ** ; mask byte count (per window width)
6170 0000F0E6 01DF              <1>       add    edi, ebx ; next row in video memory window
6171 0000F0E8 01CE              <1>       add    esi, ecx ; next window row in user's buffer
6172 0000F0EA EBB3              <1>       jmp    short sysvideo_15_141
6173                            <1> sysvideo_15_150:
6174 0000F0EC 5F                <1>       pop    edi ; ***** ; cur row in video memory window
6175 0000F0ED 5B                <1>       pop    ebx ; **** ; window width
6176 0000F0EE 5E                <1>       pop    esi ; *** ; user's buffer address
6177 0000F0EF 59                <1>       pop    ecx ; ** ; mask byte count
6178 0000F0F0 E9F7E7FFFF        <1>       jmp    sysret
6179                            <1>
6180                            <1> mask_op_sub_tbl_8:
6181 0000F0F5 [15F10000]        <1>       dd     mask_op_new_8
6182 0000F0F9 [18F10000]        <1>       dd     mask_op_add_8
6183 0000F0FD [28F10000]        <1>       dd     mask_op_avg_8
6184 0000F101 [20F10000]        <1>       dd     mask_op_sub_8
6185 0000F105 [2DF10000]        <1>       dd     mask_op_inc_8
6186 0000F109 [34F10000]        <1>       dd     mask_op_dec_8
6187 0000F10D [3BF10000]        <1>       dd     mask_op_not_8
6188 0000F111 [3EF10000]        <1>       dd     mask_op_neg_8
6189                            <1>
6190                            <1> mask_op_new_8:
6191 0000F115 881F              <1>       mov    [edi], bl
6192 0000F117 C3                <1>       retn
6193                            <1>
6194                            <1> mask_op_add_8:
6195 0000F118 001F              <1>       add    [edi], bl
6196 0000F11A 7303              <1>       jnc    short mask_op_add_8_retn
6197 0000F11C C607FF            <1>       mov    byte [edi], 0FFh
6198                            <1> mask_op_add_8_retn:
6199 0000F11F C3                <1>       retn
6200                            <1>
6201                            <1> mask_op_sub_8:
6202 0000F120 281F              <1>       sub    [edi], bl
6203 0000F122 7303              <1>       jnb    short mask_op_sub_8_retn
6204 0000F124 C60700            <1>       mov    byte [edi], 0
6205                            <1> mask_op_sub_8_retn:
6206 0000F127 C3                <1>       retn
6207                            <1>
6208                            <1> mask_op_avg_8:
6209 0000F128 001F              <1>       add    [edi], bl
6210 0000F12A D01F              <1>       rcr    byte [edi], 1
6211 0000F12C C3                <1>       retn
6212                            <1>
6213                            <1> mask_op_inc_8:
6214 0000F12D FE07              <1>       inc    byte [edi]
6215 0000F12F 7902              <1>       jns    short mask_op_inc_8_retn
6216 0000F131 FE0F              <1>       dec    byte [edi] ; 0FFh
6217                            <1> mask_op_inc_8_retn:
6218 0000F133 C3                <1>       retn
6219                            <1>
6220                            <1> mask_op_dec_8:
6221 0000F134 FE0F              <1>       dec    byte [edi]
6222 0000F136 75E7              <1>       jnz    short mask_op_add_8_retn
6223 0000F138 FE07              <1>       inc    byte [edi] ; 0
6224                            <1> mask_op_dec_8_retn:
6225 0000F13A C3                <1>       retn
6226                            <1>
6227                            <1> mask_op_not_8:
6228 0000F13B F617              <1>       not    byte [edi]
6229 0000F13D C3                <1>       retn
6230                            <1>
6231                            <1> mask_op_neg_8:
6232 0000F13E F61F              <1>       neg    byte [edi]
6233 0000F140 C3                <1>       retn
6234                            <1>
6235                            <1> sysvideo_15_151:
6236                            <1>       ; 02/01/2020
6237                            <1>       ; bh = 16 (function: bh = 2)
6238                            <1>       ; 01/01/2021
6239 0000F141 80FB0E            <1>       cmp    bl, 14
6240                            <1>       ;ja    sysvideo_15_190 ; bl = 15
6241 0000F144 0F85AB030000      <1>       jne    sysvideo_15_190 ; bl = 15
6242                            <1>
6243                            <1>       ; BL = 14 = masked write of pixels
6244                            <1>       ;      (16 bpp)
6245                            <1>
6246                            <1>       ; eax = LFB video page byte count
6247                            <1>       ;      (2 * pixel count)
6248                            <1>
6249                            <1>       ; esi = user's mask buffer address
6250                            <1>       ; edi = mask color
6251                            <1>
6252                            <1>       ; ECX Low 16 bits = Top left column (X1 position)
6253                            <1>       ; ECX High 16 bits = Top row (Y1 position)
6254                            <1>       ; EDX Low 16 bits = Bottom right column (X2 position)
6255                            <1>       ; EDX High 16 bits = Bottom row (Y2 position)
6256                            <1>
6257                            <1>       ; EBX bit 16 to 18 -> mask operation/option
6258                            <1>       ;      0 = new color (in edi)
6259                            <1>       ;      1 = add color (in edi) -up to 0FFFFh-
6260                            <1>       ;      2 = average color (in edi) -(current+edi)/2-
6261                            <1>       ;      3 = sub color (in edi) -down to 0-
6262                            <1>       ;      4 = inc      color -up to 0FFFFh-
6263                            <1>       ;      5 = dec      color -down to 0-
```

```
6264                                    <1>     ;       6 = not color
6265                                    <1>     ;       7 = neg color
6266                                    <1>
6267 0000F14A 893D[7E120300]            <1>     mov     [maskcolor], edi
6268                                    <1>
6269 0000F150 C1EB10                    <1>     shr     ebx, 16
6270 0000F153 6683E307                  <1>     and     bx, 7 ; mask option
6271 0000F157 C0E302                    <1>     shl     bl, 2 ; * 4
6272 0000F15A 8B9B[0EF20000]            <1>     mov     ebx, [ebx+mask_op_sub_tbl_16]
6273                                    <1>
6274                                    <1>     ; ebp = screen width
6275                                    <1>     ; ebx = function address
6276                                    <1>
6277 0000F160 5F                        <1>     pop     edi ; * ; LFB address
6278                                    <1>
6279                                    <1>     ; esi = user buffer (in user's memory space)
6280                                    <1>     ;       (contains mask bits, 8 * window size)
6281                                    <1>
6282 0000F161 39CA                      <1>     cmp     edx, ecx ; bottom-right > top-left ?
6283 0000F163 7310                      <1>             jnb     short sysvideo_15_152 ; window
6284                                    <1>
6285 0000F165 09D2                      <1>     or      edx, edx
6286                                    <1>     ;jnz    sysret  ; invalid !
6287 0000F167 755C                      <1>     jnz     short sysvideo_15_155
6288                                    <1>
6289                                    <1>     ; full screen masked write ; edx = 0, ecx > 0
6290                                    <1>     ; eax = LFB video page size
6291 0000F169 89C2                      <1>     mov     edx, eax
6292 0000F16B 4A                        <1>     dec     edx
6293 0000F16C 4A                        <1>     dec     edx ; last word of LFB video page
6294 0000F16D 01FA                      <1>     add     edx, edi
6295                                    <1>     ; edx = stop address
6296 0000F16F 89E9                      <1>     mov     ecx, ebp ; screen width
6297 0000F171 87DD                      <1>     xchg    ebx, ebp
6298                                    <1>     ; ebx = screen width
6299                                    <1>     ; ebp = function address
6300 0000F173 EB37                      <1>     jmp     short sysvideo_15_153
6301                                    <1>
6302                                    <1> sysvideo_15_152:
6303                                    <1>     ; masked window write
6304                                    <1>     ;cmp    dx, bp ; end column < screen width ?
6305                                    <1>     ;jb     short sysvideo_15_152_0
6306                                    <1>     ; fix bottom right position of the window
6307                                    <1>     ;mov    dx, bp
6308                                    <1>     ;dec    dx
6309 0000F175 E8AF090000               <1>     call    sysvideo_15_200
6310                                    <1> ;sysvideo_15_152_0:
6311 0000F17A 6639CA                    <1>     cmp     dx, cx
6312 0000F17D 7246                      <1>     jb      short sysvideo_15_155
6313                                    <1>
6314 0000F17F 53                        <1>     push    ebx ; (call) sub-function address
6315                                    <1>
6316 0000F180 01F8                      <1>     add     eax, edi
6317                                    <1>
6318 0000F182 50                        <1>     push    eax ; the last word of LFB + 2
6319 0000F183 57                        <1>     push    edi ; LFB start/base address
6320                                    <1>
6321 0000F184 0FB7C2                    <1>     movzx   eax, dx ; bottom right column
6322 0000F187 6629C8                    <1>     sub     ax, cx  ; - top left column
6323 0000F18A 40                        <1>     inc     eax ; same column no == 1 column
6324                                    <1>
6325 0000F18B 50                        <1>     push    eax ; pixel count per window row
6326                                    <1>
6327 0000F18C 52                        <1>     push    edx
6328                                    <1>     ;;movzx      ebx, word [ebp+2] ; screen width
6329                                    <1>     ;mov    ebx, ebp ; screen width
6330                                    <1>     ;mov    eax, ecx
6331                                    <1>     ;shr    eax, 16      ; top row
6332                                    <1>     ;mul    ebx
6333                                    <1>     ;mov    dx, cx ; top left column
6334                                    <1>     ;add    eax, edx
6335 0000F18D E8A2090000               <1>     call    sysvideo_15_202
6336 0000F192 D1E0                      <1>     shl     eax, 1 ; from pixel to byte
6337 0000F194 01C7                      <1>     add     edi, eax ; start address
6338 0000F196 59                        <1>     pop     ecx ; edx
6339                                    <1>
6340                                    <1>     ;mov    eax, ecx
6341                                    <1>     ;shr    eax, 16 ; bottom row
6342                                    <1>     ;mul    ebx
6343                                    <1>     ;mov    dx, cx ; bottom right column
6344                                    <1>     ;add    eax, edx
6345                                    <1>     ; 30/12/2020
6346                                    <1>     ; calculate end offset
6347 0000F197 E89A090000               <1>     call    sysvideo_15_203
6348 0000F19C D1E0                      <1>     shl     eax, 1 ; from pixel to byte
6349                                    <1>
6350 0000F19E 5A                        <1>     pop     edx ; pixel count per window row
6351 0000F19F 59                        <1>     pop     ecx ; LFB start/base address
6352                                    <1>
6353 0000F1A0 01C8                      <1>     add     eax, ecx ; stop address
6354 0000F1A2 59                        <1>     pop     ecx ; the last word of LFB + 2
6355                                    <1>
6356 0000F1A3 5D                        <1>     pop     ebp ; (call) sub-function address
6357                                    <1>
6358 0000F1A4 39C8                      <1>     cmp     eax, ecx
6359 0000F1A6 735D                      <1>     jnb     short sysvideo_15_163 ; wrong pos !
6360                                    <1>
6361                                    <1>     ;push   edi ; start address
6362                                    <1>
6363 0000F1A8 89D1                      <1>     mov     ecx, edx ; pixel count
6364 0000F1AA 89C2                      <1>     mov     edx, eax ; stop address
6365                                    <1>                      ; (the last word included)
6366                                    <1> sysvideo_15_153:
6367                                    <1>     ; ebp = (masked) color change function address
6368                                    <1>     ; esi = user's buffer address (virtual)
```

```
6369                                    <1>       ; edi = window start address in LFB
6370                                    <1>       ; ebx = screen width
6371                                    <1>       ; edx = stop address
6372                                    <1>       ; ecx = window width in pixels
6373 0000F1AC 66D1E3                   <1>       shl    bx, 1 ; convert pixel count to byte count
6374 0000F1AF 83C107                   <1>       add    ecx, 7
6375 0000F1B2 C1E903                   <1>       shr    ecx, 3 ; mask byte count (for user's buffer)
6376                                    <1> sysvideo_15_154:
6377                                    <1>       ; edx = stop address (the last word included!)
6378 0000F1B5 51                       <1>       push   ecx ; **
6379 0000F1B6 57                       <1>       push   edi ; * ; start address in LFB
6380                                    <1>       ; ecx = byte count (per row)
6381 0000F1B7 BF00600900               <1>       mov    edi, VBE3STACKADDR ; destination
6382                                    <1>       ; user to system video/display page window transfer
6383                                    <1>       ; esi =      user buffer
6384                                    <1>       ; edi = system buffer
6385 0000F1BC E84D210000               <1>       call   transfer_from_user_buffer ; fast transfer
6386                                    <1>              ; eax & ecx modified, other regs preserved
6387 0000F1C1 5F                       <1>       pop    edi ; * ; start address in LFB
6388 0000F1C2 7306                     <1>       jnc    short sysvideo_15_156
6389                                    <1>       ; memory error !
6390 0000F1C4 59                       <1>       pop    ecx ; **
6391                                    <1> sysvideo_15_155:
6392 0000F1C5 E922E7FFFF               <1>       jmp    sysret
6393                                    <1> sysvideo_15_156:
6394 0000F1CA 56                       <1>       push   esi ; *** ; user's buffer address
6395 0000F1CB BE00600900               <1>       mov    esi, VBE3STACKADDR ; source (in sys mem)
6396                                    <1> sysvideo_15_157:
6397 0000F1D0 53                       <1>       push   ebx ; **** ; screen width
6398 0000F1D1 57                       <1>       push   edi ; ***** ; window start address in LFB
6399 0000F1D2 668B1D[7E120300]         <1>       mov    bx, [maskcolor]
6400                                    <1> sysvideo_15_158:
6401 0000F1D9 B408                     <1>       mov    ah, 8 ; 8 mask bits for 8 pixels (16 bytes)
6402 0000F1DB AC                       <1>       lodsb
6403                                    <1> sysvideo_15_159:
6404 0000F1DC D0E8                     <1>       shr    al, 1
6405 0000F1DE 7402                     <1>       jz     short sysvideo_15_160
6406                                    <1>       ;mov   [edi], bx  ; color
6407 0000F1E0 FFD5                     <1>       call   ebp ; sub function (color change function)
6408                                    <1> sysvideo_15_160:
6409 0000F1E2 8305[64030300]02         <1>       add    dword [u.r0], 2
6410 0000F1E9 47                       <1>       inc    edi
6411 0000F1EA 47                       <1>       inc    edi
6412 0000F1EB 39D7                     <1>       cmp    edi, edx ; stop address ?
6413 0000F1ED 7716                     <1>       ja     short sysvideo_15_163
6414 0000F1EF FECC                     <1>       dec    ah
6415 0000F1F1 7405                     <1>       jz     short sysvideo_15_161
6416 0000F1F3 49                       <1>       dec    ecx
6417 0000F1F4 75E6                     <1>       jnz    short sysvideo_15_159
6418 0000F1F6 EB03                     <1>       jmp    short sysvideo_15_162
6419                                    <1> sysvideo_15_161:
6420 0000F1F8 49                       <1>       dec    ecx
6421 0000F1F9 75DE                     <1>       jnz    short sysvideo_15_158
6422                                    <1> sysvideo_15_162:
6423 0000F1FB 5F                       <1>       pop    edi ; ***** ; cur row in video memory window
6424 0000F1FC 5B                       <1>       pop    ebx ; **** ; screen width
6425 0000F1FD 5E                       <1>       pop    esi ; ***  ; user's buffer address
6426 0000F1FE 59                       <1>       pop    ecx ; ** ; mask byte count (per window width)
6427 0000F1FF 01DF                     <1>       add    edi, ebx ; next row in video memory window
6428 0000F201 01CE                     <1>       add    esi, ecx ; next window row in user's buffer
6429 0000F203 EBB0                     <1>       jmp    short sysvideo_15_154
6430                                    <1> sysvideo_15_163:
6431 0000F205 5F                       <1>       pop    edi ; ***** ; cur row in video memory window
6432 0000F206 5B                       <1>       pop    ebx ; **** ; window width
6433 0000F207 5E                       <1>       pop    esi ; *** ; user's buffer address
6434 0000F208 59                       <1>       pop    ecx ; ** ; mask byte count
6435 0000F209 E9DEE6FFFF               <1>       jmp    sysret
6436                                    <1>
6437                                    <1> mask_op_sub_tbl_16:
6438 0000F20E [2EF20000]               <1>       dd     mask_op_new_16
6439 0000F212 [32F20000]               <1>       dd     mask_op_add_16
6440 0000F216 [48F20000]               <1>       dd     mask_op_avg_16
6441 0000F21A [3DF20000]               <1>       dd     mask_op_sub_16
6442 0000F21E [4FF20000]               <1>       dd     mask_op_inc_16
6443 0000F222 [58F20000]               <1>       dd     mask_op_dec_16
6444 0000F226 [61F20000]               <1>       dd     mask_op_not_16
6445 0000F22A [65F20000]               <1>       dd     mask_op_neg_16
6446                                    <1>
6447                                    <1> mask_op_new_16:
6448 0000F22E 66891F                   <1>       mov    [edi], bx
6449 0000F231 C3                       <1>       retn
6450                                    <1>
6451                                    <1> mask_op_add_16:
6452 0000F232 66011F                   <1>       add    [edi], bx
6453 0000F235 7305                     <1>       jnc    short mask_op_add_16_retn
6454 0000F237 66C707FFFF               <1>       mov    word [edi], 0FFFFh
6455                                    <1> mask_op_add_16_retn:
6456 0000F23C C3                       <1>       retn
6457                                    <1>
6458                                    <1> mask_op_sub_16:
6459 0000F23D 66291F                   <1>       sub    [edi], bx
6460 0000F240 7305                     <1>       jnb    short mask_op_sub_16_retn
6461 0000F242 66C7070000               <1>       mov    word [edi], 0
6462                                    <1> mask_op_sub_16_retn:
6463 0000F247 C3                       <1>       retn
6464                                    <1>
6465                                    <1> mask_op_avg_16:
6466 0000F248 66011F                   <1>       add    [edi], bx
6467 0000F24B 66D11F                   <1>       rcr    word [edi], 1
6468 0000F24E C3                       <1>       retn
6469                                    <1>
6470                                    <1> mask_op_inc_16:
6471 0000F24F 66FF07                   <1>       inc    word [edi]
6472 0000F252 7503                     <1>       jnz    short mask_op_inc_16_retn
6473 0000F254 66FF0F                   <1>       dec    word [edi] ; 0FFFFh
```

```
                                <1> mask_op_inc_16_retn:
0000F257 C3                     <1>         retn
                                <1>
                                <1> mask_op_dec_16:
0000F258 66FF0F                 <1>         dec     word [edi]
0000F25B 79DF                   <1>         jns     short mask_op_add_16_retn
0000F25D 66FF07                 <1>         inc     word [edi] ; 0
                                <1> mask_op_dec_16_retn:
0000F260 C3                     <1>         retn
                                <1>
                                <1> mask_op_not_16:
0000F261 66F717                 <1>         not     word [edi]
0000F264 C3                     <1>         retn
                                <1>
                                <1> mask_op_neg_16:
0000F265 66F71F                 <1>         neg     word [edi]
0000F268 C3                     <1>         retn
                                <1>
                                <1> sysvideo_15_164:
                                <1>         ; 02/01/2020
                                <1>         ; bh = 32 (function: bh = 2)
                                <1>         ; 01/01/2021
0000F269 80FB0E                 <1>         cmp     bl, 14
                                <1>         ;ja     sysvideo_15_190 ; bl = 15
0000F26C 0F8583020000           <1>         jne     sysvideo_15_190 ; bl = 15
                                <1>
                                <1>         ; BL = 14 = masked write of pixels
                                <1>         ;       (32 bpp)
                                <1>
                                <1>         ; eax = LFB video page byte count
                                <1>         ;       (4 * pixel count)
                                <1>
                                <1>         ; esi = user's mask buffer address
                                <1>         ; edi = mask color
                                <1>
                                <1>         ; ECX Low 16 bits = Top left column (X1 position)
                                <1>         ; ECX High 16 bits = Top row (Y1 position)
                                <1>         ; EDX Low 16 bits = Bottom right column (X2 position)
                                <1>         ; EDX High 16 bits = Bottom row (Y2 position)
                                <1>
                                <1>         ; EBX bit 16 to 18 -> mask operation/option
                                <1>         ;       0 = new color (in edi)
                                <1>         ;       1 = add color (in edi) -up to 0FFFFFFFFh-
                                <1>         ;       2 = average color (in edi) -(current+edi)/2-
                                <1>         ;       3 = sub color (in edi) -down to 0-
                                <1>         ;       4 = inc       color -up to 0FFFFFFFFh-
                                <1>         ;       5 = dec       color -down to 0-
                                <1>         ;       6 = not color
                                <1>         ;       7 = neg color
                                <1>
0000F272 893D[7E120300]         <1>         mov     [maskcolor], edi
                                <1>
0000F278 C1EB10                 <1>         shr     ebx, 16
0000F27B 6683E307               <1>         and     bx, 7 ; mask option
0000F27F C0E302                 <1>         shl     bl, 2 ; * 4
0000F282 8B9B[3AF30000]         <1>         mov     ebx, [ebx+mask_op_sub_tbl_32]
                                <1>
                                <1>         ; ebp = screen width
                                <1>         ; ebx = function address
                                <1>
0000F288 5F                     <1>         pop     edi ; * ; LFB address
                                <1>
                                <1>         ; esi = user buffer (in user's memory space)
                                <1>         ;       (contains mask bits, 8 * window size)
                                <1>
0000F289 39CA                   <1>         cmp     edx, ecx ; bottom-right > top-left ?
0000F28B 7311                   <1>         jnb     short sysvideo_15_165 ; window
                                <1>
0000F28D 09D2                   <1>         or      edx, edx
                                <1>         ;jnz    sysret  ; invalid !
0000F28F 7560                   <1>         jnz     short sysvideo_15_168
                                <1>
                                <1>         ; full screen masked write ; edx = 0, ecx > 0
                                <1>         ; eax = LFB video page size
0000F291 89C2                   <1>         mov     edx, eax
0000F293 83EA04                 <1>         sub     edx, 4 ; last dword of LFB video page
0000F296 01FA                   <1>         add     edx, edi
                                <1>         ; edx = stop address
0000F298 89E9                   <1>         mov     ecx, ebp ; screen width
0000F29A 87DD                   <1>         xchg    ebx, ebp
                                <1>         ; ebx = screen width
                                <1>         ; ebp = function address
0000F29C EB39                   <1>         jmp     short sysvideo_15_166
                                <1>
                                <1> sysvideo_15_165:
                                <1>         ; masked window write
                                <1>         ;cmp    dx, bp ; end column < screen width ?
                                <1>         ;jb     short sysvideo_15_165_0
                                <1>         ; fix bottom right position of the window
                                <1>         ;mov    dx, bp
                                <1>         ;dec    dx
0000F29E E886080000             <1>         call    sysvideo_15_200
                                <1> ;sysvideo_15_165_0:
0000F2A3 6639CA                 <1>         cmp     dx, cx
0000F2A6 7249                   <1>         jb      short sysvideo_15_168
                                <1>
0000F2A8 53                     <1>         push    ebx ; (call) sub-function address
                                <1>
0000F2A9 01F8                   <1>         add     eax, edi
                                <1>
0000F2AB 50                     <1>         push    eax ; the last dword of LFB + 4
0000F2AC 57                     <1>         push    edi ; LFB start/base address
                                <1>
0000F2AD 0FB7C2                 <1>         movzx   eax, dx ; bottom right column
0000F2B0 6629C8                 <1>         sub     ax, cx  ; - top left column
```

```
6579 0000F2B3 40                <1>       inc    eax ; same column no == 1 column
6580                            <1>
6581 0000F2B4 50                <1>       push   eax ; pixel count per window row
6582                            <1>
6583 0000F2B5 52                <1>       push   edx
6584                            <1>       ;;movzx      ebx, word [ebp+2] ; screen width
6585                            <1>       ;mov   ebx, ebp ; screen width
6586                            <1>       ;mov   eax, ecx
6587                            <1>       ;shr   eax, 16      ; top row
6588                            <1>       ;mul   ebx
6589                            <1>       ;mov   dx, cx ; top left column
6590                            <1>       ;add   eax, edx
6591 0000F2B6 E879080000        <1>       call   sysvideo_15_202
6592 0000F2BB C1E002            <1>       shl    eax, 2 ; from pixel to byte
6593 0000F2BE 01C7              <1>       add    edi, eax ; start address
6594 0000F2C0 59                <1>       pop    ecx ; edx
6595                            <1>
6596                            <1>       ;mov   eax, ecx
6597                            <1>       ;shr   eax, 16 ; bottom row
6598                            <1>       ;mul   ebx
6599                            <1>       ;mov   dx, cx ; bottom right column
6600                            <1>       ;add   eax, edx
6601                            <1>       ; 30/12/2020
6602                            <1>       ; calculate end offset
6603 0000F2C1 E870080000        <1>       call   sysvideo_15_203
6604 0000F2C6 C1E002            <1>       shl    eax, 2 ; from pixel to byte
6605                            <1>
6606 0000F2C9 5A                <1>       pop    edx ; pixel count per window row
6607 0000F2CA 59                <1>       pop    ecx ; LFB start/base address
6608                            <1>
6609 0000F2CB 01C8              <1>       add    eax, ecx ; stop address
6610 0000F2CD 59                <1>       pop    ecx ; the last dword of LFB + 4
6611                            <1>
6612 0000F2CE 5D                <1>       pop    ebp ; (call) sub-function address
6613                            <1>
6614 0000F2CF 39C8              <1>       cmp    eax, ecx
6615 0000F2D1 735E              <1>       jnb    short sysvideo_15_176 ; wrong pos !
6616                            <1>
6617                            <1>       ;push  edi ; start address
6618                            <1>
6619 0000F2D3 89D1              <1>       mov    ecx, edx ; pixel count
6620 0000F2D5 89C2              <1>       mov    edx, eax ; stop address
6621                            <1>                     ; (the last dword included)
6622                            <1> sysvideo_15_166:
6623                            <1>       ; ebp = (masked) color change function address
6624                            <1>       ; esi = user's buffer address (virtual)
6625                            <1>       ; edi = window start address in LFB
6626                            <1>       ; ebx = screen width
6627                            <1>       ; edx = stop address
6628                            <1>       ; ecx = window width in pixels
6629 0000F2D7 66C1E302          <1>       shl    bx, 2 ; convert pixel count to byte count
6630 0000F2DB 83C107            <1>       add    ecx, 7
6631 0000F2DE C1E903            <1>       shr    ecx, 3 ; mask byte count (for user's buffer)
6632                            <1> sysvideo_15_167:
6633                            <1>       ; edx = stop address (the last dword included!)
6634 0000F2E1 51                <1>       push   ecx ; **
6635 0000F2E2 57                <1>       push   edi ; * ; start address in LFB
6636                            <1>       ; ecx = byte count (per row)
6637 0000F2E3 BF00600900        <1>       mov    edi, VBE3STACKADDR ; destination
6638                            <1>       ; user to system video/display page window transfer
6639                            <1>       ; esi =      user buffer
6640                            <1>       ; edi = system buffer
6641 0000F2E8 E821200000        <1>       call   transfer_from_user_buffer ; fast transfer
6642                            <1>              ; eax & ecx modified, other regs preserved
6643 0000F2ED 5F                <1>       pop    edi ; * ; start address in LFB
6644 0000F2EE 7306              <1>       jnc    short sysvideo_15_169
6645                            <1>       ; memory error !
6646 0000F2F0 59                <1>       pop    ecx ; **
6647                            <1> sysvideo_15_168:
6648 0000F2F1 E9F6E5FFFF        <1>       jmp    sysret
6649                            <1> sysvideo_15_169:
6650 0000F2F6 56                <1>       push   esi ; *** ; user's buffer address
6651 0000F2F7 BE00600900        <1>       mov    esi, VBE3STACKADDR ; source (in sys mem)
6652                            <1> sysvideo_15_170:
6653 0000F2FC 53                <1>       push   ebx ; **** ; screen width
6654 0000F2FD 57                <1>       push   edi ; ***** ; window start address in LFB
6655 0000F2FE 8B1D[7E120300]    <1>       mov    ebx, [maskcolor]
6656                            <1> sysvideo_15_171:
6657 0000F304 B408              <1>       mov    ah, 8 ; 8 mask bits for 8 pixels (32 bytes)
6658 0000F306 AC                <1>       lodsb
6659                            <1> sysvideo_15_172:
6660 0000F307 D0E8              <1>       shr    al, 1
6661 0000F309 7402              <1>       jz     short sysvideo_15_173
6662                            <1>       ;mov   [edi], ebx  ; color
6663 0000F30B FFD5              <1>       call   ebp ; sub function (color change function)
6664                            <1> sysvideo_15_173:
6665 0000F30D 8305[64030300]04  <1>       add    dword [u.r0], 4
6666 0000F314 83C704            <1>       add    edi, 4
6667 0000F317 39D7              <1>       cmp    edi, edx ; stop address ?
6668 0000F319 7716              <1>       ja     short sysvideo_15_176
6669 0000F31B FECC              <1>       dec    ah
6670 0000F31D 7405              <1>       jz     short sysvideo_15_174
6671 0000F31F 49                <1>       dec    ecx
6672 0000F320 75E5              <1>       jnz    short sysvideo_15_172
6673 0000F322 EB03              <1>       jmp    short sysvideo_15_175
6674                            <1> sysvideo_15_174:
6675 0000F324 49                <1>       dec    ecx
6676 0000F325 75DD              <1>       jnz    short sysvideo_15_171
6677                            <1> sysvideo_15_175:
6678 0000F327 5F                <1>       pop    edi ; ***** ; cur row in video memory window
6679 0000F328 5B                <1>       pop    ebx ; **** ; screen width
6680 0000F329 5E                <1>       pop    esi ; ***  ; user's buffer address
6681 0000F32A 59                <1>       pop    ecx ; ** ; mask byte count (per window width)
6682 0000F32B 01DF              <1>       add    edi, ebx ; next row in video memory window
6683 0000F32D 01CE              <1>       add    esi, ecx ; next window row in user's buffer
```

```
6684 0000F32F EBB0              <1>       jmp    short sysvideo_15_167
6685                            <1> sysvideo_15_176:
6686 0000F331 5F                <1>       pop    edi ; ***** ; cur row in video memory window
6687 0000F332 5B                <1>       pop    ebx ; **** ; window width
6688 0000F333 5E                <1>       pop    esi ; *** ; user's buffer address
6689 0000F334 59                <1>       pop    ecx ; ** ; mask byte count
6690 0000F335 E9B2E5FFFF        <1>       jmp    sysret
6691                            <1>
6692                            <1> mask_op_sub_tbl_32:
6693 0000F33A [5AF30000]        <1>       dd     mask_op_new_32
6694 0000F33E [5DF30000]        <1>       dd     mask_op_add_32
6695 0000F342 [73F30000]        <1>       dd     mask_op_avg_32
6696 0000F346 [68F30000]        <1>       dd     mask_op_sub_32
6697 0000F34A [78F30000]        <1>       dd     mask_op_inc_32
6698 0000F34E [7FF30000]        <1>       dd     mask_op_dec_32
6699 0000F352 [86F30000]        <1>       dd     mask_op_not_32
6700 0000F356 [89F30000]        <1>       dd     mask_op_neg_32
6701                            <1>
6702                            <1> mask_op_new_32:
6703 0000F35A 891F              <1>       mov    [edi], ebx
6704 0000F35C C3                <1>       retn
6705                            <1>
6706                            <1> mask_op_add_32:
6707 0000F35D 011F              <1>       add    [edi], ebx
6708 0000F35F 7306              <1>       jnc    short mask_op_add_32_retn
6709 0000F361 C707FFFFFFFF      <1>       mov    dword [edi], 0FFFFFFFFh
6710                            <1> mask_op_add_32_retn:
6711 0000F367 C3                <1>       retn
6712                            <1>
6713                            <1> mask_op_sub_32:
6714 0000F368 291F              <1>       sub    [edi], ebx
6715 0000F36A 7306              <1>       jnb    short mask_op_sub_32_retn
6716 0000F36C C70700000000      <1>       mov    dword [edi], 0
6717                            <1> mask_op_sub_32_retn:
6718 0000F372 C3                <1>       retn
6719                            <1>
6720                            <1> mask_op_avg_32:
6721 0000F373 011F              <1>       add    [edi], ebx
6722 0000F375 D11F              <1>       rcr    dword [edi], 1
6723 0000F377 C3                <1>       retn
6724                            <1>
6725                            <1> mask_op_inc_32:
6726 0000F378 FF07              <1>       inc    dword [edi]
6727 0000F37A 7502              <1>       jnz    short mask_op_inc_32_retn
6728 0000F37C FF0F              <1>       dec    dword [edi] ; 0FFFFFFFFh
6729                            <1> mask_op_inc_32_retn:
6730 0000F37E C3                <1>       retn
6731                            <1>
6732                            <1> mask_op_dec_32:
6733 0000F37F FF0F              <1>       dec    dword [edi]
6734 0000F381 79E4              <1>       jns    short mask_op_add_32_retn
6735 0000F383 FF07              <1>       inc    dword [edi]
6736                            <1> mask_op_dec_32_retn:
6737 0000F385 C3                <1>       retn
6738                            <1>
6739                            <1> mask_op_not_32:
6740 0000F386 F717              <1>       not    dword [edi]
6741 0000F388 C3                <1>       retn
6742                            <1>
6743                            <1> mask_op_neg_32:
6744 0000F389 F71F              <1>       neg    dword [edi]
6745 0000F38B C3                <1>       retn
6746                            <1>
6747                            <1> sysvideo_15_177:
6748                            <1>       ; 02/01/2020
6749                            <1>       ; bh = 24 (function: bh = 2)
6750                            <1>       ; 01/01/2021
6751 0000F38C 80FB0E            <1>       cmp    bl, 14
6752                            <1>       ;ja    sysvideo_15_190 ; bl = 15
6753 0000F38F 0F8560010000      <1>       jne    sysvideo_15_190 ; bl = 15
6754                            <1>
6755                            <1>       ; BL = 14 = masked write of pixels
6756                            <1>       ;      (24 bpp)
6757                            <1>
6758                            <1>       ; eax = LFB video page byte count
6759                            <1>       ;      (3 * pixel count)
6760                            <1>
6761                            <1>       ; esi = user's mask buffer address
6762                            <1>       ; edi = mask color
6763                            <1>
6764                            <1>       ; ECX Low 16 bits = Top left column (X1 position)
6765                            <1>       ; ECX High 16 bits = Top row (Y1 position)
6766                            <1>       ; EDX Low 16 bits = Bottom right column (X2 position)
6767                            <1>       ; EDX High 16 bits = Bottom row (Y2 position)
6768                            <1>
6769                            <1>       ; EBX bit 16 to 18 -> mask operation/option
6770                            <1>       ;      0 = new color (in edi)
6771                            <1>       ;      1 = add color (in edi) -up to 0FFFFFFFFh-
6772                            <1>       ;      2 = average color (in edi) -(current+edi)/2-
6773                            <1>       ;      3 = sub color (in edi) -down to 0-
6774                            <1>       ;      4 = inc      color -up to 0FFFFFFFFh-
6775                            <1>       ;      5 = dec      color -down to 0-
6776                            <1>       ;      6 = not color
6777                            <1>       ;      7 = neg color
6778                            <1>
6779 0000F395 81E7FFFFFF00      <1>       and    edi, 0FFFFFFh
6780 0000F39B 893D[7E120300]    <1>       mov    [maskcolor], edi
6781                            <1>
6782 0000F3A1 C1EB10            <1>       shr    ebx, 16
6783 0000F3A4 6683E307          <1>       and    bx, 7 ; mask option
6784 0000F3A8 C0E302            <1>       shl    bl, 2 ; * 4
6785 0000F3AB 8B9B[6BF40000]    <1>       mov    ebx, [ebx+mask_op_sub_tbl_24]
6786                            <1>
6787                            <1>       ; ebp = screen width
6788                            <1>       ; ebx = function address
```

```
                                <1>
0000F3B1 5F                     <1>       pop    edi ; * ; LFB address
                                <1>
                                <1>       ; esi = user buffer (in user's memory space)
                                <1>       ;       (contains mask bits, 8 * window size)
                                <1>
0000F3B2 39CA                   <1>       cmp    edx, ecx ; bottom-right > top-left ?
0000F3B4 7311                   <1>              jnb    short sysvideo_15_178 ; window
                                <1>
0000F3B6 09D2                   <1>       or     edx, edx
                                <1>       ;jnz   sysret  ; invalid !
0000F3B8 7568                   <1>       jnz    short sysvideo_15_181
                                <1>
                                <1>       ; full screen masked write ; edx = 0, ecx > 0
                                <1>       ; eax = LFB video page size
0000F3BA 89C2                   <1>       mov    edx, eax
0000F3BC 83EA03                 <1>       sub    edx, 3 ; last 3 bytes of LFB video page
0000F3BF 01FA                   <1>       add    edx, edi
                                <1>       ; edx = stop address
0000F3C1 89E9                   <1>       mov    ecx, ebp ; screen width
0000F3C3 87DD                   <1>       xchg   ebx, ebp
                                <1>       ; ebx = screen width
                                <1>       ; ebp = function address
0000F3C5 EB3F                   <1>       jmp    short sysvideo_15_179
                                <1>
                                <1> sysvideo_15_178:
                                <1>       ; masked window write
                                <1>       ;cmp   dx, bp ; end column < screen width ?
                                <1>       ;jb    short sysvideo_15_178_0
                                <1>       ; fix bottom right position of the window
                                <1>       ;mov   dx, bp
                                <1>       ;dec   dx
0000F3C7 E85D070000             <1>       call   sysvideo_15_200
                                <1> ;sysvideo_15_178_0:
0000F3CC 6639CA                 <1>       cmp    dx, cx
0000F3CF 7251                   <1>       jb     short sysvideo_15_181
                                <1>
0000F3D1 53                     <1>       push   ebx ; (call) sub-function address
                                <1>
0000F3D2 01F8                   <1>       add    eax, edi
                                <1>
0000F3D4 50                     <1>       push   eax ; the last 3 bytes of LFB + 3
0000F3D5 57                     <1>       push   edi ; LFB start/base address
                                <1>
0000F3D6 0FB7C2                 <1>       movzx  eax, dx ; bottom right column
0000F3D9 6629C8                 <1>       sub    ax, cx  ; - top left column
0000F3DC 40                     <1>       inc    eax ; same column no == 1 column
                                <1>
0000F3DD 50                     <1>       push   eax ; pixel count per window row
                                <1>
0000F3DE 52                     <1>       push   edx
                                <1>       ;;movzx      ebx, word [ebp+2] ; screen width
                                <1>       ;mov   ebx, ebp ; screen width
                                <1>       ;mov   eax, ecx
                                <1>       ;shr   eax, 16      ; top row
                                <1>       ;mul   ebx
                                <1>       ;mov   dx, cx ; top left column
                                <1>       ;add   eax, edx
0000F3DF E850070000             <1>       call   sysvideo_15_202
0000F3E4 89C1                   <1>       mov    ecx, eax
0000F3E6 D1E0                   <1>       shl    eax, 1
0000F3E8 01C8                   <1>       add    eax, ecx ; from pixel to byte
0000F3EA 01C7                   <1>       add    edi, eax ; start address
0000F3EC 59                     <1>       pop    ecx ; edx
                                <1>
                                <1>       ;mov   eax, ecx
                                <1>       ;shr   eax, 16 ; bottom row
                                <1>       ;mul   ebx
                                <1>       ;mov   dx, cx ; bottom right column
                                <1>       ;add   eax, edx
                                <1>       ; 30/12/2020
                                <1>       ; calculate end offset
0000F3ED E844070000             <1>       call   sysvideo_15_203
0000F3F2 89C2                   <1>       mov    edx, eax
0000F3F4 D1E0                   <1>       shl    eax, 1
0000F3F6 01D0                   <1>       add    eax, edx ; from pixel to byte
                                <1>
0000F3F8 5A                     <1>       pop    edx ; pixel count per window row
0000F3F9 59                     <1>       pop    ecx ; LFB start/base address
                                <1>
0000F3FA 01C8                   <1>       add    eax, ecx ; stop address
0000F3FC 59                     <1>       pop    ecx ; the last 3 bytes of LFB + 3
                                <1>
0000F3FD 5D                     <1>       pop    ebp ; (call) sub-function address
                                <1>
0000F3FE 39C8                   <1>       cmp    eax, ecx
0000F400 7360                   <1>       jnb    short sysvideo_15_189 ; wrong pos !
                                <1>
                                <1>       ;push  edi ; start address
                                <1>
0000F402 89D1                   <1>       mov    ecx, edx ; pixel count
0000F404 89C2                   <1>       mov    edx, eax ; stop address
                                <1>                       ; (the last 3 bytes included)
                                <1> sysvideo_15_179:
                                <1>       ; ebp = (masked) color change function address
                                <1>       ; esi = user's buffer address (virtual)
                                <1>       ; edi = window start address in LFB
                                <1>       ; ebx = screen width
                                <1>       ; edx = stop address
                                <1>       ; ecx = window width in pixels
0000F406 89D8                   <1>       mov    eax, ebx
0000F408 01C3                   <1>       add    ebx, eax
0000F40A 01C3                   <1>       add    ebx, eax ; convert pixel count to byte count
0000F40C 83C107                 <1>       add    ecx, 7
0000F40F C1E903                 <1>       shr    ecx, 3 ; mask byte count (for user's buffer)
```

```
                                <1> sysvideo_15_180:
                                <1>       ; edx = stop address (the last 3 bytes included!)
0000F412 51                     <1>       push   ecx ; **
0000F413 57                     <1>       push   edi ; * ; start address in LFB
                                <1>       ; ecx = byte count (per row)
0000F414 BF00600900             <1>       mov    edi, VBE3STACKADDR ; destination
                                <1>       ; user to system video/display page window transfer
                                <1>       ; esi =      user buffer
                                <1>       ; edi = system buffer
0000F419 E8F01E0000             <1>       call   transfer_from_user_buffer ; fast transfer
                                <1>              ; eax & ecx modified, other regs preserved
0000F41E 5F                     <1>       pop    edi ; * ; start address in LFB
0000F41F 7306                   <1>       jnc    short sysvideo_15_182
                                <1>       ; memory error !
0000F421 59                     <1>       pop    ecx ; **
                                <1> sysvideo_15_181:
0000F422 E9C5E4FFFF             <1>       jmp    sysret
                                <1> sysvideo_15_182:
0000F427 56                     <1>       push   esi ; *** ; user's buffer address
0000F428 BE00600900             <1>       mov    esi, VBE3STACKADDR ; source (in sys mem)
                                <1> sysvideo_15_183:
0000F42D 53                     <1>       push   ebx ; **** ; screen width
0000F42E 57                     <1>       push   edi ; ***** ; window start address in LFB
0000F42F 8B1D[7E120300]         <1>       mov    ebx, [maskcolor]
                                <1> sysvideo_15_184:
0000F435 B408                   <1>       mov    ah, 8 ; 8 mask bits for 8 pixels (24 bytes)
0000F437 AC                     <1>       lodsb
                                <1> sysvideo_15_185:
0000F438 D0E8                   <1>       shr    al, 1
0000F43A 7402                   <1>       jz     short sysvideo_15_186
                                <1>       ;mov   [edi], ebx  ; color
0000F43C FFD5                   <1>       call   ebp ; sub function (color change function)
                                <1> sysvideo_15_186:
0000F43E 8305[64030300]03       <1>       add    dword [u.r0], 3
0000F445 83C703                 <1>       add    edi, 3
0000F448 39D7                   <1>       cmp    edi, edx ; stop address ?
0000F44A 7716                   <1>       ja     short sysvideo_15_189
0000F44C FECC                   <1>       dec    ah
0000F44E 7405                   <1>       jz     short sysvideo_15_187
0000F450 49                     <1>       dec    ecx
0000F451 75E5                   <1>       jnz    short sysvideo_15_185
0000F453 EB03                   <1>       jmp    short sysvideo_15_188
                                <1> sysvideo_15_187:
0000F455 49                     <1>       dec    ecx
0000F456 75DD                   <1>       jnz    short sysvideo_15_184
                                <1> sysvideo_15_188:
0000F458 5F                     <1>       pop    edi ; ***** ; cur row in video memory window
0000F459 5B                     <1>       pop    ebx ; **** ; screen width
0000F45A 5E                     <1>       pop    esi ; ***  ; user's buffer address
0000F45B 59                     <1>       pop    ecx ; ** ; mask byte count (per window width)
0000F45C 01DF                   <1>       add    edi, ebx ; next row in video memory window
0000F45E 01CE                   <1>       add    esi, ecx ; next window row in user's buffer
0000F460 EBB0                   <1>       jmp    short sysvideo_15_180
                                <1> sysvideo_15_189:
0000F462 5F                     <1>       pop    edi ; ***** ; cur row in video memory window
0000F463 5B                     <1>       pop    ebx ; **** ; window width
0000F464 5E                     <1>       pop    esi ; *** ; user's buffer address
0000F465 59                     <1>       pop    ecx ; ** ; mask byte count
0000F466 E981E4FFFF             <1>       jmp    sysret
                                <1>
                                <1> mask_op_sub_tbl_24:
0000F46B [8BF40000]             <1>       dd     mask_op_new_24
0000F46F [94F40000]             <1>       dd     mask_op_add_24
0000F473 [BEF40000]             <1>       dd     mask_op_avg_24
0000F477 [AEF40000]             <1>       dd     mask_op_sub_24
0000F47B [CCF40000]             <1>       dd     mask_op_inc_24
0000F47F [D9F40000]             <1>       dd     mask_op_dec_24
0000F483 [E7F40000]             <1>       dd     mask_op_not_24
0000F487 [EEF40000]             <1>       dd     mask_op_neg_24
                                <1>
                                <1> mask_op_new_24:
0000F48B 8127000000FF           <1>       and    dword [edi], 0FF000000h
0000F491 091F                   <1>       or     dword [edi], ebx ; ebx <= 0FFFFFFh
                                <1>       ;add   dword [edi], ebx ; ebx <= 0FFFFFFh
0000F493 C3                     <1>       retn
                                <1>
                                <1> mask_op_add_24:
0000F494 50                     <1>       push   eax
0000F495 8B07                   <1>       mov    eax, [edi]
0000F497 0D000000FF             <1>       or     eax, 0FF000000h
0000F49C 01D8                   <1>       add    eax, ebx
0000F49E 7303                   <1>       jnc    short mask_op_add_24_0
0000F4A0 31C0                   <1>       xor    eax, eax
                                <1> mask_op_inc_24_1:
0000F4A2 48                     <1>       dec    eax ; 0FFFFFFFFh
                                <1> mask_op_add_24_0:
                                <1> mask_op_sub_24_0:
                                <1> mask_op_avg_24_0:
                                <1> mask_op_inc_24_0:
                                <1> mask_op_dec_24_0:
                                <1> mask_op_not_24_0:
                                <1> mask_op_neg_24_0:
0000F4A3 668907                 <1>       mov    [edi], ax
0000F4A6 C1E810                 <1>       shr    eax, 16
0000F4A9 884702                 <1>       mov    [edi+2], al
0000F4AC 58                     <1>       pop    eax
0000F4AD C3                     <1>       retn
                                <1>
                                <1> mask_op_sub_24:
0000F4AE 50                     <1>       push   eax
0000F4AF 8B07                   <1>       mov    eax, [edi]
0000F4B1 25FFFFFF00             <1>       and    eax, 0FFFFFFh
0000F4B6 29D8                   <1>       sub    eax, ebx
0000F4B8 73E9                   <1>       jnb    short mask_op_sub_24_0
0000F4BA 31C0                   <1>       xor    eax, eax ; 0
```

```
6999 0000F4BC EBE5              <1>       jmp    short mask_op_sub_24_0
7000                            <1> ;mask_op_sub_24_0:
7001                            <1>       ;mov   [edi], ax
7002                            <1>       ;shr   eax, 16
7003                            <1>       ;mov   [edi+2], al
7004                            <1>       ;pop   eax
7005                            <1>       ;retn
7006                            <1>
7007                            <1> mask_op_avg_24:
7008 0000F4BE 50                <1>       push   eax
7009 0000F4BF 8B07              <1>       mov    eax, [edi]
7010 0000F4C1 25FFFFFF00        <1>       and    eax, 0FFFFFFh
7011 0000F4C6 01D8              <1>       add    eax, ebx
7012 0000F4C8 D1D8              <1>       rcr    eax, 1
7013 0000F4CA EBD7              <1>       jmp    short mask_op_avg_24_0
7014                            <1>       ;mov   [edi], ax
7015                            <1>       ;shr   eax, 16
7016                            <1>       ;mov   [edi+2], al
7017                            <1>       ;pop   eax
7018                            <1>       ;retn
7019                            <1>
7020                            <1> mask_op_inc_24:
7021 0000F4CC 50                <1>       push   eax
7022 0000F4CD 8B07              <1>       mov    eax, [edi]
7023 0000F4CF 0D000000FF        <1>       or     eax, 0FF000000h
7024 0000F4D4 40                <1>       inc    eax
7025 0000F4D5 75CC              <1>       jnz    short mask_op_inc_24_0
7026                            <1>       ;dec   eax ; 0FFFFFFFFh
7027 0000F4D7 EBC9              <1>       jmp    short mask_op_inc_24_1
7028                            <1> ;mask_op_inc_24_0:
7029                            <1>       ;mov   [edi], ax
7030                            <1>       ;shr   eax, 16
7031                            <1>       ;mov   [edi+2], al
7032                            <1>       ;pop   eax
7033                            <1>       ;retn
7034                            <1>
7035                            <1> mask_op_dec_24:
7036 0000F4D9 50                <1>       push   eax
7037 0000F4DA 8B07              <1>       mov    eax, [edi]
7038 0000F4DC 25FFFFFF00        <1>       and    eax, 0FFFFFFh
7039 0000F4E1 48                <1>       dec    eax
7040 0000F4E2 79BF              <1>       jns    short mask_op_dec_24_0
7041 0000F4E4 40                <1>       inc    eax ; eax = 0
7042 0000F4E5 EBBC              <1>       jmp    short mask_op_dec_24_0
7043                            <1> ;mask_op_dec_24_0:
7044                            <1>       ;mov   [edi], ax
7045                            <1>       ;shr   eax, 16
7046                            <1>       ;mov   [edi+2], al
7047                            <1>       ;pop   eax
7048                            <1>       ;retn
7049                            <1>
7050                            <1> mask_op_not_24:
7051 0000F4E7 50                <1>       push   eax
7052 0000F4E8 8B07              <1>       mov    eax, [edi]
7053 0000F4EA F7D0              <1>       not    eax
7054 0000F4EC EBB5              <1>       jmp    short mask_op_not_24_0
7055                            <1> ;mask_op_not_24_0:
7056                            <1>       ;mov   [edi], ax
7057                            <1>       ;shr   eax, 16
7058                            <1>       ;mov   [edi+2], al
7059                            <1>       ;pop   eax
7060                            <1>       ;retn
7061                            <1>
7062                            <1> mask_op_neg_24:
7063 0000F4EE 50                <1>       push   eax
7064 0000F4EF 8B07              <1>       mov    eax, [edi]
7065                            <1>       ;and   eax, 0FFFFFFh
7066 0000F4F1 F7D8              <1>       neg    eax
7067 0000F4F3 EBAE              <1>       jmp    short mask_op_neg_24_0
7068                            <1> ;mask_op_neg_24_0:
7069                            <1>       ;mov   [edi], ax
7070                            <1>       ;shr   eax, 16
7071                            <1>       ;mov   [edi+2], al
7072                            <1>       ;pop   eax
7073                            <1>       ;retn
7074                            <1>
7075                            <1> sysvideo_15_190:
7076                            <1>       ; 05/01/2021 ([ufont])
7077                            <1>       ; 01/01/2021
7078                            <1>       ; BL = 15 -> Write character (font)
7079                            <1>       ;            -in current mode-
7080                            <1>       ; ECX Low 16 bits = Top left column (X1 position)
7081                            <1>       ; ECX High 16 bits = Top row (Y1 position)
7082                            <1>       ;  DL = Character's ASCII code
7083                            <1>       ; EDX High 24 bits = Character's color
7084                            <1>       ; EBX bit 16 -> character size option
7085                            <1>       ;     0 = 8x16 system character font
7086                            <1>       ;     1 = 8x8 system character font
7087                            <1>       ; EBX bit 17 & 18 -> scale
7088                            <1>       ;     0 = 1/1 (8 pixels per char row)
7089                            <1>       ;     1 = 2/1 (16 pixels per char row)
7090                            <1>       ;     2 = 4/1 (32 pixels per char row)
7091                            <1>       ;     3 = 8/1 (64 pixels per char row)
7092                            <1>       ; EBX bit 19 -> user font option (1 = user font)
7093                            <1>
7094                            <1>       ; eax = LFB video page byte count
7095                            <1>       ; ebp = screen width
7096                            <1>
7097                            <1>       ;  bh = bits per pixel (8,16,24,32)
7098                            <1>
7099 0000F4F5 5F                <1>       pop    edi ; * ; LFB address
7100                            <1>
7101 0000F4F6 8915[7E120300]    <1>       mov    [maskcolor], edx ; save as temporary
7102                            <1>
7103 0000F4FC 53                <1>       push   ebx  ; + ; save bits per pixel value
```

```
7104                                    <1>
7105                                    <1>     ; calculate scaled character size
7106 0000F4FD 89DA                      <1>     mov   edx, ebx
7107 0000F4FF C1EA10                    <1>     shr   edx, 16
7108 0000F502 6683FA0F                  <1>     cmp   dx, 15 ; 05/01/2021
7109 0000F506 7605                      <1>     jna   short sysvideo_15_191
7110                                    <1>     ; invalid
7111 0000F508 E9DFE3FFFF                <1>     jmp   sysret
7112                                    <1> sysvideo_15_191:
7113                                    <1>     ; 05/01/2021
7114 0000F50D 8025[82120300]FC          <1>     and   byte [ufont], ~3 ; bit 0&1 = 0
7115 0000F514 F6C208                    <1>     test  dl, 8 ; user font ?
7116 0000F517 742A                      <1>     jz    short sysvideo_15_191_2 ; use sys font
7117 0000F519 80E207                    <1>     and   dl, 7 ; clear bit 19
7118 0000F51C F6C201                    <1>     test  dl, 1
7119 0000F51F 7412                      <1>     jz    short sysvideo_15_191_0 ; 8x16
7120                                    <1>     ; 8x8
7121 0000F521 F605[82120300]08          <1>     test  byte [ufont], 8 ; is 8x8 user font ready ?
7122 0000F528 7419                      <1>     jz    short sysvideo_15_191_2
7123                                    <1>                           ; no, use system font
7124                                    <1>     ; yes
7125 0000F52A 800D[82120300]01          <1>     or    byte [ufont], 1 ; bit 0 = 8x8 pixels char
7126 0000F531 EB10                      <1>     jmp   short sysvideo_15_191_2
7127                                    <1> sysvideo_15_191_0:
7128 0000F533 F605[82120300]10          <1>     test  byte [ufont], 16 ; is 8x16 user font ready ?
7129 0000F53A 7407                      <1>     jz    short sysvideo_15_191_2
7130                                    <1>                           ; no, use system font
7131                                    <1>     ; yes
7132 0000F53C 800D[82120300]02          <1>     or    byte [ufont], 2 ; bit 1 = 8x16 pixels char
7133                                    <1> sysvideo_15_191_2:
7134 0000F543 BE[61F80000]              <1>     mov   esi, char_size_option
7135 0000F548 D0E2                      <1>     shl   dl, 1
7136 0000F54A 01D6                      <1>     add   esi, edx
7137 0000F54C 668B16                    <1>     mov   dx, [esi] ; dh = rows, dl = pixels
7138 0000F54F C1E208                    <1>     shl   edx, 8
7139 0000F552 86F2                      <1>     xchg  dh, dl ; dh = 0, dl = columns
7140                                    <1>           ; EDX high word = rows
7141                                    <1>           ; DX = columns (pixels per row)
7142                                    <1>
7143 0000F554 52                        <1>     push  edx ; ++
7144 0000F555 6601CA                    <1>     add   dx, cx ; dx = bottom right column
7145 0000F558 7307                      <1>     jnc   short sysvideo_15_193
7146                                    <1> sysvideo_15_192:
7147 0000F55A 5A                        <1>     pop   edx ; ++
7148 0000F55B 5B                        <1>     pop   ebx ; +
7149 0000F55C E98BE3FFFF                <1>     jmp   sysret
7150                                    <1> sysvideo_15_193:
7151 0000F561 6639EA                    <1>     cmp   dx, bp ; dx > screen width ?
7152 0000F564 77F4                      <1>     ja    short sysvideo_15_192 ; yes, wrong pos !
7153 0000F566 C1CA10                    <1>     ror   edx, 16
7154 0000F569 C1C910                    <1>     ror   ecx, 16
7155 0000F56C 6601CA                    <1>     add   dx, cx ; dx = bottom row
7156 0000F56F 72E9                      <1>     jc    short sysvideo_15_192 ; wrong pos !
7157 0000F571 C1C210                    <1>     rol   edx, 16
7158 0000F574 C1C110                    <1>     rol   ecx, 16
7159                                    <1>
7160 0000F577 50                        <1>     push  eax ; +++  ; LFB video page size in bytes
7161 0000F578 52                        <1>     push  edx ; ++++ ; bottom row, right column
7162                                    <1>     ;mov  ebx, ebp ; screen width
7163                                    <1>     ;mov  eax, ecx
7164                                    <1>     ;shr  eax, 16  ; top row
7165                                    <1>     ;mul  ebx
7166                                    <1>     ;mov  dx, cx ; top left column
7167                                    <1>     ;add  eax, edx
7168 0000F579 E8B6050000                <1>     call  sysvideo_15_202
7169 0000F57E 59                        <1>     pop   ecx ; ++++ ; edx
7170 0000F57F 01C7                      <1>     add   edi, eax ; edi = LFB start address
7171 0000F581 3B0424                    <1>     cmp   eax, [esp]  ; < LFB page size ?
7172 0000F584 7203                      <1>     jb    short sysvideo_15_194  ; yes, proper position
7173 0000F586 58                        <1>     pop   eax ; +++
7174 0000F587 EBD1                      <1>     jmp   short sysvideo_15_192
7175                                    <1> sysvideo_15_194:
7176                                    <1>     ;mov  eax, ecx
7177                                    <1>     ;shr  eax, 16 ; bottom row
7178                                    <1>     ;mul  ebx
7179                                    <1>     ;mov  dx, cx ; bottom right column
7180                                    <1>     ;add  eax, edx
7181                                    <1>     ; 30/12/2020
7182                                    <1>     ; calculate end offset
7183 0000F589 E8A8050000                <1>     call  sysvideo_15_203
7184 0000F58E 3B0424                    <1>     cmp   eax, [esp] ; < LFB page size ?
7185 0000F591 58                        <1>     pop   eax ; +++
7186 0000F592 73C6                      <1>     jnb   short sysvideo_15_192 ; no, wrong position !
7187                                    <1>
7188                                    <1>     ; edi = character start position
7189                                    <1>     ;     (top left)
7190                                    <1>
7191                                    <1>     ; ebp = screen width
7192                                    <1>
7193 0000F594 5A                        <1>     pop   edx ; ++ ; char columns (pixels), rows
7194                                    <1>
7195 0000F595 5B                        <1>     pop   ebx ; +      ; bh = bits per pixel
7196                                    <1>
7197 0000F596 80FF08                    <1>     cmp   bh, 8
7198 0000F599 7614                      <1>     jna   short sysvideo_15_196
7199 0000F59B 89E8                      <1>     mov   eax, ebp
7200 0000F59D D1E5                      <1>     shl   ebp, 1
7201 0000F59F 80FF10                    <1>     cmp   bh, 16
7202 0000F5A2 760B                      <1>     jna   short sysvideo_15_196
7203 0000F5A4 80FF18                    <1>     cmp   bh, 24
7204 0000F5A7 7504                      <1>     jne   short sysvideo_15_195
7205 0000F5A9 01C5                      <1>     add   ebp, eax ; * 3
7206 0000F5AB EB02                      <1>     jmp   short sysvideo_15_196
7207                                    <1> sysvideo_15_195:
7208                                    <1>     ; bh = 32
```

```
7209 0000F5AD D1E5               <1>       shl    ebp, 1
7210                             <1> sysvideo_15_196:
7211                             <1>       ; ebp = screen width (converted to bytes)
7212                             <1>       ; edi = character start position in LFB
7213                             <1>       ;  bh = bytes per pixel
7214                             <1>
7215 0000F5AF A1[7E120300]       <1>       mov    eax, [maskcolor]
7216 0000F5B4 0FB6F0             <1>       movzx  esi, al  ; character (ASCII code)
7217 0000F5B7 C1E808             <1>       shr    eax, 8
7218 0000F5BA A3[7E120300]       <1>       mov    [maskcolor], eax
7219                             <1>
7220 0000F5BF 0FB7C2             <1>       movzx  eax, dx ; columns (pixels)
7221 0000F5C2 C1EA10             <1>       shr    edx, 16 ; rows
7222                             <1>
7223                             <1>       ; edi = LFB start address (top left column)
7224                             <1>
7225 0000F5C5 39D0               <1>       cmp    eax, edx ; 8x8 or 8x16 font
7226 0000F5C7 0F846B010000       <1>       je     sysvideo_15_197 ; 8x8 font
7227                             <1>
7228                             <1>       ; 8x16 character font
7229 0000F5CD 66C1E604           <1>       shl    si, 4 ; * 16 (character font size)
7230                             <1>
7231                             <1>       ; 05/01/2021
7232                             <1>             ; test 8x16 user font is ready flag
7233 0000F5D1 F605[82120300]02   <1>       test   byte [ufont], 2
7234 0000F5D8 7408               <1>       jz     short sysvideo_15_196_1
7235                             <1>       ; 8x16 user font available
7236 0000F5DA 81C600400900       <1>       add    esi, VGAFONT16USER
7237 0000F5E0 EB06               <1>       jmp    short sysvideo_15_196_2
7238                             <1> sysvideo_15_196_1:
7239 0000F5E2 81C6[246E0100]     <1>       add    esi, vgafont16
7240                             <1> sysvideo_15_196_2:
7241 0000F5E8 8B15[7E120300]     <1>       mov    edx, [maskcolor]
7242                             <1>
7243 0000F5EE B110               <1>       mov    cl, 16 ; 16 rows (8x16 font)
7244                             <1>
7245 0000F5F0 80FF08             <1>       cmp    bh, 8  ; 8bpp ?
7246 0000F5F3 7754               <1>       ja     short sysvideo_15_196_22 ; no
7247                             <1>
7248 0000F5F5 3C08               <1>       cmp    al, 8  ; 8 pixels per row ?
7249 0000F5F7 7711               <1>       ja     short sysvideo_15_196_4   ; no
7250                             <1>
7251                             <1>       ; 8 pixels per row
7252                             <1> ;sysvideo_15_196_0:
7253                             <1> ;     lodsb
7254                             <1> ;     push   edi
7255                             <1> ;     mov    ah, 8
7256                             <1> ;sysvideo_15_196_1:
7257                             <1> ;     shr    al, 1
7258                             <1> ;     jnc    short sysvideo_15_196_2
7259                             <1> ;     mov    [edi], dl
7260                             <1> ;            ; character pixel/forecolor,
7261                             <1> ;            ; 8 bpp (256 colors)
7262                             <1> ;sysvideo_15_196_2:
7263                             <1> ;     inc    edi
7264                             <1> ;     dec    ah
7265                             <1> ;     jnz    short sysvideo_15_196_1 ; next pixel of (same) row
7266                             <1> ;     pop    edi
7267                             <1> ;     add    edi, ebp ; next row (same column position)
7268                             <1> ;     dec    cl
7269                             <1> ;     jnz    short sysvideo_15_196_0
7270                             <1> ;                       ; next row of the character font
7271 0000F5F9 E873020000         <1>       call   sysvideo_15_196_0
7272                             <1>
7273                             <1> sysvideo_15_196_3:
7274                             <1>       ; 1*16*1 = 16 bytes (128 pixels)
7275 0000F5FE C605[64030300]10   <1>       mov    byte [u.r0], 16 ; font size = 16 bytes
7276 0000F605 E9E2E2FFFF         <1>       jmp    sysret
7277                             <1>
7278                             <1> sysvideo_15_196_4:
7279 0000F60A 3C10               <1>       cmp    al, 16
7280 0000F60C 7711               <1>       ja     short sysvideo_15_196_10 ; >= 32 pixels per row
7281                             <1>       ; 16 pixels per row
7282                             <1> ;sysvideo_15_196_4_0:
7283                             <1> ;     mov    dh, dl
7284                             <1> ;sysvideo_15_196_5:
7285                             <1> ;     mov    bl, 2  ; 2 rows with same pixels
7286                             <1> ;sysvideo_15_196_6:
7287                             <1> ;     mov    al, [esi]
7288                             <1> ;     push   edi
7289                             <1> ;     mov    ah, 8
7290                             <1> ;sysvideo_15_196_7:
7291                             <1> ;     shr    al, 1
7292                             <1> ;     jnc    short sysvideo_15_196_8
7293                             <1> ;     ; 2 pixels with same forecolor
7294                             <1> ;     mov    [edi], dx
7295                             <1> ;            ; character pixel/forecolor,
7296                             <1> ;            ; 8 bpp (256 colors)
7297                             <1> ;sysvideo_15_196_8:
7298                             <1> ;     inc    edi
7299                             <1> ;     inc    edi
7300                             <1> ;     dec    ah
7301                             <1> ;     jnz    short sysvideo_15_196_7 ; next pixel of (same) row
7302                             <1> ;     pop    edi
7303                             <1> ;     add    edi, ebp ; next row
7304                             <1> ;     dec    bl
7305                             <1> ;     jnz    short sysvideo_15_196_6 ; same pixels for next row
7306                             <1> ;     inc    esi
7307                             <1> ;     dec    cl
7308                             <1> ;     jnz    short sysvideo_15_196_5 ; next row of the character font
7309 0000F60E E875020000         <1>       call   sysvideo_15_196_4_0
7310                             <1>
7311                             <1> sysvideo_15_196_9:
7312                             <1>       ; 2*16*2 = 64 bytes (512 pixels)
7313 0000F613 C605[64030300]40   <1>       mov    byte [u.r0], 64 ; font size = 64 bytes
```

```
7314 0000F61A E9CDE2FFFF          <1>       jmp    sysret
7315                              <1>
7316                              <1> sysvideo_15_196_10:
7317 0000F61F 3C20                <1>       cmp    al, 32
7318 0000F621 7713                <1>       ja     short sysvideo_15_196_16 ; 64 pixels per row
7319                              <1>       ; 32 pixels per row
7320                              <1> ;sysvideo_15_196_10_0:
7321                              <1> ;     mov    dh, dl
7322                              <1> ;     ror    edx, 8  ; byte 2 -> dh, dl -> byte 3
7323                              <1> ;                    ; dh -> dl
7324                              <1> ;     mov    dh, dl ; byte 1 -> dl
7325                              <1> ;sysvideo_15_196_11:
7326                              <1> ;     mov    bl, 4  ; 4 rows with same pixels
7327                              <1> ;sysvideo_15_196_12:
7328                              <1> ;     mov    al, [esi]
7329                              <1> ;     push   edi
7330                              <1> ;     mov    ah, 8
7331                              <1> ;sysvideo_15_196_13:
7332                              <1> ;     shr    al, 1
7333                              <1> ;     jnc    short sysvideo_15_196_14
7334                              <1> ;     ; 4 pixels with same forecolor
7335                              <1> ;     mov    [edi], edx
7336                              <1> ;            ; character pixel/forecolor,
7337                              <1> ;            ; 8 bpp (256 colors)
7338                              <1> ;sysvideo_15_196_14:
7339                              <1> ;     add    edi, 4
7340                              <1> ;     dec    ah
7341                              <1> ;     jnz    short sysvideo_15_196_13 ; next pixel of (same) row
7342                              <1> ;     pop    edi
7343                              <1> ;     add    edi, ebp ; next row
7344                              <1> ;     dec    bl
7345                              <1> ;     jnz    short sysvideo_15_196_12 ; same pixels for next row
7346                              <1> ;     inc    esi
7347                              <1> ;     dec    cl
7348                              <1> ;     jnz    short sysvideo_15_196_11 ; next row of the character font
7349 0000F623 E883020000          <1>       call   sysvideo_15_196_10_0
7350                              <1>
7351                              <1> sysvideo_15_196_15:
7352                              <1>       ; 4*16*4 = 256 bytes (2048 pixels)
7353 0000F628 66C705[64030300]00- <1>       mov    word [u.r0], 256 ; font size = 256 bytes
7353 0000F630 01                  <1>
7354 0000F631 E9B6E2FFFF          <1>       jmp    sysret
7355                              <1>
7356                              <1> sysvideo_15_196_16:
7357                              <1> ;     ; 64 pixels per row
7358                              <1> ;     mov    dh, dl
7359                              <1> ;     ror    edx, 8  ; byte 2 -> dh, dl -> byte 3
7360                              <1> ;                    ; dh -> dl
7361                              <1> ;     mov    dh, dl ; byte 1 -> dl
7362                              <1> ;sysvideo_15_196_17:
7363                              <1> ;     mov    bl, 8  ; 8 rows with same pixels
7364                              <1> ;sysvideo_15_196_18:
7365                              <1> ;     mov    al, [esi]
7366                              <1> ;     push   edi
7367                              <1> ;     mov    ah, 8
7368                              <1> ;sysvideo_15_196_19:
7369                              <1> ;     shr    al, 1
7370                              <1> ;     jnc    short sysvideo_15_196_20
7371                              <1> ;     ; 8 pixels with same forecolor
7372                              <1> ;     mov    [edi], edx
7373                              <1> ;            ; character pixel/forecolor,
7374                              <1> ;            ; 8 bpp (256 colors)
7375                              <1> ;     mov    [edi+4], edx
7376                              <1> ;            ; character pixel/forecolor,
7377                              <1> ;            ; 8 bpp (256 colors)
7378                              <1> ;sysvideo_15_196_20:
7379                              <1> ;     add    edi, 8
7380                              <1> ;     dec    ah
7381                              <1> ;     jnz    short sysvideo_15_196_19 ; next pixel of (same) row
7382                              <1> ;     pop    edi
7383                              <1> ;     add    edi, ebp ; next row
7384                              <1> ;     dec    bl
7385                              <1> ;     jnz    short sysvideo_15_196_18 ; same pixels for next row
7386                              <1> ;     inc    esi
7387                              <1> ;     dec    cl
7388                              <1> ;     jnz    short sysvideo_15_196_17 ; next row of the character font
7389 0000F636 E898020000          <1>       call   sysvideo_15_196_16_0
7390                              <1>
7391                              <1> sysvideo_15_196_21:
7392                              <1>       ; 8*16*8 = 1024 bytes (8192 pixels)
7393 0000F63B 66C705[64030300]00- <1>       mov    word [u.r0], 1024 ; font size = 1024 bytes
7393 0000F643 04                  <1>
7394 0000F644 E9A3E2FFFF          <1>       jmp    sysret
7395                              <1>
7396                              <1> sysvideo_15_196_22:
7397 0000F649 80FF10              <1>       cmp    bh, 16 ; 16bpp ?
7398 0000F64C 7754                <1>       ja     short sysvideo_15_196_45 ; no
7399                              <1>
7400 0000F64E 3C08                <1>       cmp    al, 8  ; 8 pixels per row ?
7401 0000F650 7711                <1>       ja     short sysvideo_15_196_27 ; no
7402                              <1>
7403                              <1>       ; 8 pixels per row
7404                              <1> ;sysvideo_15_196_23:
7405                              <1> ;     lodsb
7406                              <1> ;     push   edi
7407                              <1> ;     mov    ah, 8
7408                              <1> ;sysvideo_15_196_24:
7409                              <1> ;     shr    al, 1
7410                              <1> ;     jnc    short sysvideo_15_196_25
7411                              <1> ;     mov    [edi], dx
7412                              <1> ;            ; character pixel/forecolor,
7413                              <1> ;            ; 16 bpp (65536 colors)
7414                              <1> ;sysvideo_15_196_25:
7415                              <1> ;     inc    edi
7416                              <1> ;     inc    edi
```

```
7417                                 <1> ;       dec     ah
7418                                 <1> ;       jnz     short sysvideo_15_196_24 ; next pixel of (same) row
7419                                 <1> ;       pop     edi
7420                                 <1> ;       add     edi, ebp ; next row (same column position)
7421                                 <1> ;       dec     cl
7422                                 <1> ;       jnz     short sysvideo_15_196_23
7423                                 <1>                         ; next row of the character font
7424 0000F652 E8A7020000            <1>         call    sysvideo_15_196_23
7425                                 <1>
7426                                 <1> sysvideo_15_196_26:
7427                                 <1>         ; 1*16*1*2 = 32 bytes (128 pixels)
7428 0000F657 C605[64030300]20       <1>         mov     byte [u.r0], 32 ; font size = 32 bytes
7429 0000F65E E989E2FFFF             <1>         jmp     sysret
7430                                 <1>
7431                                 <1> sysvideo_15_196_27:
7432 0000F663 3C10                   <1>         cmp     al, 16
7433 0000F665 7711                   <1>         ja      short sysvideo_15_196_33 ; >= 32 pixels per row
7434                                 <1> ;sysvideo_15_196_27_0:
7435                                 <1> ;       ; 16 pixels per row
7436                                 <1> ;       ; (4 bytes per row)
7437                                 <1> ;       mov     bx, dx
7438                                 <1> ;       shl     edx, 16
7439                                 <1> ;       mov     dx, bx
7440                                 <1> ;sysvideo_15_196_28:
7441                                 <1> ;       mov     bl, 2  ; 2 rows with same pixels
7442                                 <1> ;sysvideo_15_196_29:
7443                                 <1> ;       mov     al, [esi]
7444                                 <1> ;       push    edi
7445                                 <1> ;       mov     ah, 8
7446                                 <1> ;sysvideo_15_196_30:
7447                                 <1> ;       shr     al, 1
7448                                 <1> ;       jnc     short sysvideo_15_196_31
7449                                 <1> ;       ; 2 pixels with same forecolor
7450                                 <1> ;       mov     [edi], edx
7451                                 <1> ;               ; character pixel/forecolor,
7452                                 <1> ;               ; 16 bpp (65536 colors)
7453                                 <1> ;sysvideo_15_196_31:
7454                                 <1> ;       add     edi, 4
7455                                 <1> ;       dec     ah
7456                                 <1> ;       jnz     short sysvideo_15_196_30 ; next pixel of (same) row
7457                                 <1> ;       pop     edi
7458                                 <1> ;       add     edi, ebp ; next row
7459                                 <1> ;       dec     bl
7460                                 <1> ;       jnz     short sysvideo_15_196_29 ; same pixels for next row
7461                                 <1> ;       inc     esi
7462                                 <1> ;       dec     cl
7463                                 <1> ;       jnz     short sysvideo_15_196_28 ; next row of the character font
7464 0000F667 E8AB020000            <1>         call    sysvideo_15_196_27_0
7465                                 <1>
7466                                 <1> sysvideo_15_196_32:
7467                                 <1>         ; 2*16*2*2 = 128 bytes (512 pixels)
7468 0000F66C C605[64030300]80       <1>         mov     byte [u.r0], 128 ; font size = 128 bytes
7469 0000F673 E974E2FFFF             <1>         jmp     sysret
7470                                 <1>
7471                                 <1> sysvideo_15_196_33:
7472 0000F678 3C20                   <1>         cmp     al, 32
7473 0000F67A 7713                   <1>         ja      short sysvideo_15_196_39 ; 64 pixels per row
7474                                 <1> ;sysvideo_15_196_33_0:
7475                                 <1> ;       ; 32 pixels per row
7476                                 <1> ;       ; (8 bytes per row)
7477                                 <1> ;       mov     bx, dx
7478                                 <1> ;       shl     edx, 16
7479                                 <1> ;       mov     dx, bx
7480                                 <1> ;sysvideo_15_196_34:
7481                                 <1> ;       mov     bl, 4  ; 4 rows with same pixels
7482                                 <1> ;sysvideo_15_196_35:
7483                                 <1> ;       mov     al, [esi]
7484                                 <1> ;       push    edi
7485                                 <1> ;       mov     ah, 8
7486                                 <1> ;sysvideo_15_196_36:
7487                                 <1> ;       shr     al, 1
7488                                 <1> ;       jnc     short sysvideo_15_196_37
7489                                 <1> ;       ; 4 pixels with same forecolor
7490                                 <1> ;       mov     [edi], edx
7491                                 <1> ;               ; character pixel/forecolor,
7492                                 <1> ;               ; 16 bpp (65536 colors)
7493                                 <1> ;       mov     [edi+4], edx
7494                                 <1> ;               ; character pixel/forecolor,
7495                                 <1> ;               ; 16 bpp (65536 colors)
7496                                 <1> ;sysvideo_15_196_37:
7497                                 <1> ;       add     edi, 8
7498                                 <1> ;       dec     ah
7499                                 <1> ;       jnz     short sysvideo_15_196_36 ; next pixel of (same) row
7500                                 <1> ;       pop     edi
7501                                 <1> ;       add     edi, ebp ; next row
7502                                 <1> ;       dec     bl
7503                                 <1> ;       jnz     short sysvideo_15_196_35 ; same pixels for next row
7504                                 <1> ;       inc     esi
7505                                 <1> ;       dec     cl
7506                                 <1> ;       jnz     short sysvideo_15_196_34 ; next row of the character font
7507 0000F67C E8C0020000            <1>         call    sysvideo_15_196_33_0
7508                                 <1>
7509                                 <1> sysvideo_15_196_38:
7510                                 <1>         ; 4*16*4*2 = 512 bytes (2048 pixels)
7511 0000F681 66C705[64030300]00-    <1>         mov     word [u.r0], 512 ; font size = 512 bytes
7511 0000F689 02                     <1>
7512 0000F68A E95DE2FFFF             <1>         jmp     sysret
7513                                 <1>
7514                                 <1> sysvideo_15_196_39:
7515                                 <1> ;       ; 64 pixels per row
7516                                 <1> ;       ; (16 bytes per row)
7517                                 <1> ;       mov     bx, dx
7518                                 <1> ;       shl     edx, 16
7519                                 <1> ;       mov     dx, bx
7520                                 <1> ;sysvideo_15_196_40:
```

```
7521                                    <1> ;       mov     bl, 8  ; 8 rows with same pixels
7522                                    <1> ;sysvideo_15_196_41:
7523                                    <1> ;       mov     al, [esi]
7524                                    <1> ;       push    edi
7525                                    <1> ;       mov     ah, 8
7526                                    <1> ;sysvideo_15_196_42:
7527                                    <1> ;       shr     al, 1
7528                                    <1> ;       jnc     short sysvideo_15_196_43
7529                                    <1> ;       ; 8 pixels with same forecolor
7530                                    <1> ;       mov     [edi], edx
7531                                    <1> ;       mov     [edi+4], edx
7532                                    <1> ;       mov     [edi+8], edx
7533                                    <1> ;       mov     [edi+12], edx
7534                                    <1> ;               ; character pixel/forecolor,
7535                                    <1> ;               ; 16 bpp (65536 colors)
7536                                    <1> ;sysvideo_15_196_43:
7537                                    <1> ;       add     edi, 16
7538                                    <1> ;       dec     ah
7539                                    <1> ;       jnz     short sysvideo_15_196_42 ; next pixel of (same) row
7540                                    <1> ;       pop     edi
7541                                    <1> ;       add     edi, ebp ; next row
7542                                    <1> ;       dec     bl
7543                                    <1> ;       jnz     short sysvideo_15_196_41 ; same pixels for next row
7544                                    <1> ;       inc     esi
7545                                    <1> ;       dec     cl
7546                                    <1> ;       jnz     short sysvideo_15_196_40 ; next row of the character font
7547 0000F68F E8DA020000               <1>         call    sysvideo_15_196_39_0
7548                                    <1>
7549                                    <1> sysvideo_15_196_44:
7550                                    <1>         ; 8*16*8*2 = 2048 bytes (8192 pixels)
7551 0000F694 66C705[64030300]00-       <1>         mov     word [u.r0], 2048 ; font size = 2048 bytes
7551 0000F69C 08                        <1>
7552 0000F69D E94AE2FFFF                <1>         jmp     sysret
7553                                    <1>
7554                                    <1> sysvideo_15_196_45:
7555 0000F6A2 80FF20                    <1>         cmp     bh, 32 ; 16bpp ?
7556 0000F6A5 723D                      <1>         jb      short sysvideo_15_196_68 ; no
7557                                    <1>
7558 0000F6A7 3C08                      <1>         cmp     al, 8  ; 8 pixels per row ?
7559 0000F6A9 770A                      <1>         ja      short sysvideo_15_196_50 ; no
7560                                    <1>         ; 8 pixels per row
7561                                    <1> ;sysvideo_15_196_46:
7562                                    <1> ;       lodsb
7563                                    <1> ;       push    edi
7564                                    <1> ;       mov     ah, 8
7565                                    <1> ;sysvideo_15_196_47:
7566                                    <1> ;       shr     al, 1
7567                                    <1> ;       jnc     short sysvideo_15_196_48
7568                                    <1> ;       mov     [edi], edx
7569                                    <1> ;               ; character pixel/forecolor,
7570                                    <1> ;               ; 32 bpp (true colors)
7571                                    <1> ;sysvideo_15_196_48:
7572                                    <1> ;       add     edi, 4
7573                                    <1> ;       dec     ah
7574                                    <1> ;       jnz     short sysvideo_15_196_47 ; next pixel of (same) row
7575                                    <1> ;       pop     edi
7576                                    <1> ;       add     edi, ebp ; next row (same column position)
7577                                    <1> ;       dec     cl
7578                                    <1> ;       jnz     short sysvideo_15_196_46
7579                                    <1> ;                           ; next row of the character font
7580 0000F6AB E8F1020000               <1>         call    sysvideo_15_196_46
7581                                    <1>
7582                                    <1> sysvideo_15_196_49:
7583                                    <1>         ; 1*16*1*4 = 64 bytes (128 pixels)
7584                                    <1>         ;mov   byte [u.r0], 64 ; font size = 64 bytes
7585                                    <1>         ;jmp   sysret
7586                                    <1>         ; 03/01/2021
7587 0000F6B0 E95EFFFFFF               <1>         jmp     sysvideo_15_196_9
7588                                    <1>
7589                                    <1> sysvideo_15_196_50:
7590 0000F6B5 3C10                      <1>         cmp     al, 16
7591 0000F6B7 770A                      <1>         ja      short sysvideo_15_196_56 ; >= 32 pixels per row
7592                                    <1> ;sysvideo_15_196_50_0:
7593                                    <1> ;       ; 16 pixels per row
7594                                    <1> ;       ; (8 bytes per row)
7595                                    <1> ;sysvideo_15_196_51:
7596                                    <1> ;       mov     bl, 2  ; 2 rows with same pixels
7597                                    <1> ;sysvideo_15_196_52:
7598                                    <1> ;       mov     al, [esi]
7599                                    <1> ;       push    edi
7600                                    <1> ;       mov     ah, 8
7601                                    <1> ;sysvideo_15_196_53:
7602                                    <1> ;       shr     al, 1
7603                                    <1> ;       jnc     short sysvideo_15_196_54
7604                                    <1> ;       ; 2 pixels with same forecolor
7605                                    <1> ;       mov     [edi], edx
7606                                    <1> ;       mov     [edi+4], edx
7607                                    <1> ;               ; character pixel/forecolor,
7608                                    <1> ;               ; 32 bpp (true colors)
7609                                    <1> ;sysvideo_15_196_54:
7610                                    <1> ;       add     edi, 8
7611                                    <1> ;       dec     ah
7612                                    <1> ;       jnz     short sysvideo_15_196_53 ; next pixel of (same) row
7613                                    <1> ;       pop     edi
7614                                    <1> ;       add     edi, ebp ; next row
7615                                    <1> ;       dec     bl
7616                                    <1> ;       jnz     short sysvideo_15_196_52 ; same pixels for next row
7617                                    <1> ;       inc     esi
7618                                    <1> ;       dec     cl
7619                                    <1> ;       jnz     short sysvideo_15_196_51 ; next row of the character font
7620 0000F6B9 E8FC020000               <1>         call    sysvideo_15_196_50_0
7621                                    <1>
7622                                    <1> sysvideo_15_196_55:
7623                                    <1>         ; 2*16*2*4 = 256 bytes (512 pixels)
7624                                    <1>         ;mov   word [u.r0], 256 ; font size = 256 bytes
```

```
                            <1>       ;jmp   sysret
                            <1>       ; 03/01/2021
0000F6BE E965FFFFFF         <1>       jmp    sysvideo_15_196_15
                            <1>
                            <1> sysvideo_15_196_56:
0000F6C3 3C20               <1>       cmp    al, 32
0000F6C5 770A               <1>       ja     short sysvideo_15_196_62 ; 64 pixels per row
                            <1> ;sysvideo_15_196_56_0:
                            <1> ;     ; 32 pixels per row
                            <1> ;     ; (16 bytes per row)
                            <1> ;sysvideo_15_196_57:
                            <1> ;     mov    bl, 4  ; 4 rows with same pixels
                            <1> ;sysvideo_15_196_58:
                            <1> ;     mov    al, [esi]
                            <1> ;     push   edi
                            <1> ;     mov    ah, 8
                            <1> ;sysvideo_15_196_59:
                            <1> ;     shr    al, 1
                            <1> ;     jnc    short sysvideo_15_196_60
                            <1> ;     ; 4 pixels with same forecolor
                            <1> ;     mov    [edi], edx
                            <1> ;     mov    [edi+4], edx
                            <1> ;     mov    [edi+8], edx
                            <1> ;     mov    [edi+12], edx
                            <1> ;            ; character pixel/forecolor,
                            <1> ;            ; 32 bpp (true colors)
                            <1> ;sysvideo_15_196_60:
                            <1> ;     add    edi, 16
                            <1> ;     dec    ah
                            <1> ;     jnz    short sysvideo_15_196_59 ; next pixel of (same) row
                            <1> ;     pop    edi
                            <1> ;     add    edi, ebp ; next row
                            <1> ;     dec    bl
                            <1> ;     jnz    short sysvideo_15_196_58 ; same pixels for next row
                            <1> ;     inc    esi
                            <1> ;     dec    cl
                            <1> ;     jnz    short sysvideo_15_196_57 ; next row of the character font
0000F6C7 E812030000         <1>       call   sysvideo_15_196_56_0
                            <1>
                            <1> sysvideo_15_196_61:
                            <1>       ; 4*16*4*4 = 1024 bytes (2048 pixels)
                            <1>       ;mov   word [u.r0], 1024 ; font size = 1024 bytes
                            <1>       ;jmp   sysret
                            <1>       ; 03/01/2021
0000F6CC E96AFFFFFF         <1>       jmp    sysvideo_15_196_21
                            <1>
                            <1> sysvideo_15_196_62:
                            <1> ;     ; 64 pixels per row
                            <1> ;     ; (16 bytes per row)
                            <1> ;     mov    bx, dx
                            <1> ;     shl    edx, 16
                            <1> ;     mov    dx, bx
                            <1> ;sysvideo_15_196_63:
                            <1> ;     mov    bl, 8  ; 8 rows with same pixels
                            <1> ;sysvideo_15_196_64:
                            <1> ;     mov    al, [esi]
                            <1> ;     push   edi
                            <1> ;     mov    ah, 8
                            <1> ;sysvideo_15_196_65:
                            <1> ;     shr    al, 1
                            <1> ;     jnc    short sysvideo_15_196_66
                            <1> ;     ; 8 pixels with same forecolor
                            <1> ;     mov    [edi], edx
                            <1> ;     mov    [edi+4], edx
                            <1> ;     mov    [edi+8], edx
                            <1> ;     mov    [edi+12], edx
                            <1> ;     mov    [edi+16], edx
                            <1> ;     mov    [edi+20], edx
                            <1> ;     mov    [edi+24], edx
                            <1> ;     mov    [edi+28], edx
                            <1> ;            ; character pixel/forecolor,
                            <1> ;            ; 32 bpp (true colors)
                            <1> ;sysvideo_15_196_66:
                            <1> ;     add    edi, 32
                            <1> ;     dec    ah
                            <1> ;     jnz    short sysvideo_15_196_65 ; next pixel of (same) row
                            <1> ;     pop    edi
                            <1> ;     add    edi, ebp ; next row
                            <1> ;     dec    bl
                            <1> ;     jnz    short sysvideo_15_196_64 ; same pixels for next row
                            <1> ;     inc    esi
                            <1> ;     dec    cl
                            <1> ;     jnz    short sysvideo_15_196_63 ; next row of the character font
0000F6D1 E832030000         <1>       call   sysvideo_15_196_62_0
                            <1>
                            <1> sysvideo_15_196_67:
                            <1>       ; 8*16*8*4 = 4096 bytes (8192 pixels)
0000F6D6 66C705[64030300]00- <1>      mov    word [u.r0], 4096 ; font size = 4096 bytes
0000F6DE 10                 <1>
0000F6DF E908E2FFFF         <1>       jmp    sysret
                            <1>
                            <1> sysvideo_15_196_68:
                            <1>       ; bh = 24
                            <1>       ; 24 bpp
0000F6E4 3C08               <1>       cmp    al, 8  ; 8 pixels per row ?
0000F6E6 7711               <1>       ja     short sysvideo_15_196_73 ; no
                            <1>       ; 8 pixels per row
                            <1> ;sysvideo_15_196_69:
                            <1> ;     lodsb
                            <1> ;     push   edi
                            <1> ;     mov    ah, 8
                            <1> ;sysvideo_15_196_70:
                            <1> ;     shr    al, 1
                            <1> ;     jnc    short sysvideo_15_196_71
                            <1> ;     ;mov   ebx, edx
```

```
                            <1> ;       ;shr   ebx, 16
                            <1> ;       ;mov   [edi], dx
                            <1> ;       ;mov   [edi+2], bl
                            <1> ;       call   sysvideo_15_196_91
                            <1> ;              ; character pixel/forecolor,
                            <1> ;              ; 24 bpp (true colors)
                            <1> ;sysvideo_15_196_71:
                            <1> ;       add    edi, 3
                            <1> ;       dec    ah
                            <1> ;       jnz    short sysvideo_15_196_70 ; next pixel of (same) row
                            <1> ;       pop    edi
                            <1> ;       add    edi, ebp ; next row (same column position)
                            <1> ;       dec    cl
                            <1> ;       jnz    short sysvideo_15_196_69
                            <1> ;                         ; next row of the character font
0000F6E8 E85A030000         <1>         call   sysvideo_15_196_69
                            <1>
                            <1> sysvideo_15_196_72:
                            <1>         ; 1*16*1*3 = 48 bytes (128 pixels)
0000F6ED C605[64030300]30   <1>         mov    byte [u.r0], 48 ; font size = 48 bytes
0000F6F4 E9F3E1FFFF         <1>         jmp    sysret
                            <1>
                            <1> sysvideo_15_196_73:
0000F6F9 3C10               <1>         cmp    al, 16
0000F6FB 7711               <1>         ja     short sysvideo_15_196_79 ; >= 32 pixels per row
                            <1>         ; 16 pixels per row
                            <1>         ; (6 bytes per row)
                            <1> ;sysvideo_15_196_74:
                            <1> ;       mov    bl, 2  ; 2 rows with same pixels
                            <1> ;sysvideo_15_196_75:
                            <1> ;       mov    al, [esi]
                            <1> ;       push   edi
                            <1> ;       mov    ah, 8
                            <1> ;sysvideo_15_196_76:
                            <1> ;       shr    al, 1
                            <1> ;       jnc    short sysvideo_15_196_77
                            <1> ;       ; 2 pixels with same forecolor
                            <1> ;       ;mov   ebx, edx
                            <1> ;       ;shr   ebx, 16
                            <1> ;       ;mov   [edi], dx
                            <1> ;       ;mov   [edi+2], bl
                            <1> ;       call   sysvideo_15_196_91
                            <1> ;       ;mov   [edi+3], dx
                            <1> ;       ;mov   [edi+5], bl
                            <1> ;       call   sysvideo_15_196_92
                            <1> ;              ; character pixel/forecolor,
                            <1> ;              ; 24 bpp (true colors)
                            <1> ;sysvideo_15_196_77:
                            <1> ;       add    edi, 6
                            <1> ;       dec    ah
                            <1> ;       jnz    short sysvideo_15_196_76 ; next pixel of (same) row
                            <1> ;       pop    edi
                            <1> ;       add    edi, ebp ; next row
                            <1> ;       dec    bl
                            <1> ;       jnz    short sysvideo_15_196_75 ; same pixels for next row
                            <1> ;       inc    esi
                            <1> ;       dec    cl
                            <1> ;       jnz    short sysvideo_15_196_74 ; next row of the character font
0000F6FD E861030000         <1>         call   sysvideo_15_196_74
                            <1>
                            <1> sysvideo_15_196_78:
                            <1>         ; 2*16*2*3 = 192 bytes (512 pixels)
0000F702 C605[64030300]C0   <1>         mov    byte [u.r0], 192 ; font size = 192 bytes
0000F709 E9DEE1FFFF         <1>         jmp    sysret
                            <1>
                            <1> sysvideo_15_196_79:
0000F70E 3C20               <1>         cmp    al, 32
0000F710 7713               <1>         ja     short sysvideo_15_196_85 ; 64 pixels per row
                            <1>         ; 32 pixels per row
                            <1>         ; (12 bytes per row)
                            <1> ;sysvideo_15_196_80:
                            <1> ;       mov    bl, 4  ; 4 rows with same pixels
                            <1> ;sysvideo_15_196_81:
                            <1> ;       mov    al, [esi]
                            <1> ;       push   edi
                            <1> ;       mov    ah, 8
                            <1> ;sysvideo_15_196_82:
                            <1> ;       shr    al, 1
                            <1> ;       jnc    short sysvideo_15_196_83
                            <1> ;       ; 4 pixels with same forecolor
                            <1> ;       ;mov   ebx, edx
                            <1> ;       ;shr   ebx, 16
                            <1> ;       ;mov   [edi], dx
                            <1> ;       ;mov   [edi+2], bl
                            <1> ;       call   sysvideo_15_196_91
                            <1> ;       ;mov   [edi+3], dx
                            <1> ;       ;mov   [edi+5], bl
                            <1> ;       ;mov   [edi+6], dx
                            <1> ;       ;mov   [edi+8], bl
                            <1> ;       ;mov   [edi+9], dx
                            <1> ;       ;mov   [edi+11], bl
                            <1> ;       call   sysvideo_15_196_93
                            <1> ;              ; character pixel/forecolor,
                            <1> ;              ; 24 bpp (true colors)
                            <1> ;sysvideo_15_196_83:
                            <1> ;       add    edi, 12
                            <1> ;       dec    ah
                            <1> ;       jnz    short sysvideo_15_196_82 ; next pixel of (same) row
                            <1> ;       pop    edi
                            <1> ;       add    edi, ebp ; next row
                            <1> ;       dec    bl
                            <1> ;       jnz    short sysvideo_15_196_81 ; same pixels for next row
                            <1> ;       inc    esi
                            <1> ;       dec    cl
                            <1> ;       jnz    short sysvideo_15_196_80 ; next row of the character font
```

```
7834 0000F712 E875030000          <1>       call   sysvideo_15_196_80
7835                              <1>
7836                              <1> sysvideo_15_196_84:
7837                              <1>       ; 4*16*4*3 = 768 bytes (2048 pixels)
7838 0000F717 66C705[64030300]00- <1>       mov    word [u.r0], 768 ; font size = 768 bytes
7838 0000F71F 03                  <1>
7839 0000F720 E9C7E1FFFF          <1>       jmp    sysret
7840                              <1>
7841                              <1> sysvideo_15_196_85:
7842                              <1> ;     ; 64 pixels per row
7843                              <1> ;     ; (24 bytes per row)
7844                              <1> ;     mov    bx, dx
7845                              <1> ;     shl    edx, 16
7846                              <1> ;     mov    dx, bx
7847                              <1> ;sysvideo_15_196_86:
7848                              <1> ;     mov    bl, 8  ; 8 rows with same pixels
7849                              <1> ;sysvideo_15_196_87:
7850                              <1> ;     mov    al, [esi]
7851                              <1> ;     push   edi
7852                              <1> ;     mov    ah, 8
7853                              <1> ;sysvideo_15_196_88:
7854                              <1> ;     shr    al, 1
7855                              <1> ;     jnc    short sysvideo_15_196_89
7856                              <1> ;     ; 8 pixels with same forecolor
7857                              <1> ;     ;mov   ebx, edx
7858                              <1> ;     ;shr   ebx, 16
7859                              <1> ;     ;mov   [edi], dx
7860                              <1> ;     ;mov   [edi+2], bl
7861                              <1> ;     call   sysvideo_15_196_91
7862                              <1> ;     ;mov   [edi+3], dx
7863                              <1> ;     ;mov   [edi+5], bl
7864                              <1> ;     ;mov   [edi+6], dx
7865                              <1> ;     ;mov   [edi+8], bl
7866                              <1> ;     ;mov   [edi+9], dx
7867                              <1> ;     ;mov   [edi+11], bl
7868                              <1> ;     ;mov   [edi+12], dx
7869                              <1> ;     ;mov   [edi+14], bl
7870                              <1> ;     ;mov   [edi+15], dx
7871                              <1> ;     ;mov   [edi+17], bl
7872                              <1> ;     ;mov   [edi+18], dx
7873                              <1> ;     ;mov   [edi+20], bl
7874                              <1> ;     ;mov   [edi+21], dx
7875                              <1> ;     ;mov   [edi+23], bl
7876                              <1> ;     call   sysvideo_15_196_94
7877                              <1> ;            ; character pixel/forecolor,
7878                              <1> ;            ; 24 bpp (true colors)
7879                              <1> ;sysvideo_15_196_89:
7880                              <1> ;     add    edi, 24
7881                              <1> ;     dec    ah
7882                              <1> ;     jnz    short sysvideo_15_196_88 ; next pixel of (same) row
7883                              <1> ;     pop    edi
7884                              <1> ;     add    edi, ebp ; next row
7885                              <1> ;     dec    bl
7886                              <1> ;     jnz    short sysvideo_15_196_87 ; same pixels for next row
7887                              <1> ;     inc    esi
7888                              <1> ;     dec    cl
7889                              <1> ;     jnz    short sysvideo_15_196_86 ; next row of the character font
7890 0000F725 E88B030000          <1>       call   sysvideo_15_196_85_0
7891                              <1>
7892                              <1> sysvideo_15_196_90:
7893                              <1>       ; 8*16*8*3 = 3072 bytes (8192 pixels)
7894 0000F72A 66C705[64030300]00- <1>       mov    word [u.r0], 3072 ; font size = 3072 bytes
7894 0000F732 0C                  <1>
7895 0000F733 E9B4E1FFFF          <1>       jmp    sysret
7896                              <1>
7897                              <1> sysvideo_15_197:
7898                              <1>       ; 05/01/2021 ([ufont])
7899                              <1>
7900                              <1>       ; ebp = screen width (converted to bytes)
7901                              <1>       ; edi = character start position in LFB
7902                              <1>       ;  bh = bytes per pixel (video mode)
7903                              <1>       ;  al = font pixels per row
7904                              <1>
7905                              <1>       ; esi = ASCII code of the character
7906                              <1>
7907                              <1>       ; 8x8 character font
7908 0000F738 66C1E603            <1>       shl    si, 3 ; * 8 (character font size)
7909                              <1>
7910                              <1>       ; 05/01/2021
7911                              <1>             ; test 8x8 user font is ready flag
7912 0000F73C F605[82120300]01    <1>       test   byte [ufont], 1
7913 0000F743 7408                <1>       jz     short sysvideo_15_197_1
7914                              <1>       ; 8x8 user font available
7915 0000F745 81C600500900        <1>       add    esi, VGAFONT8USER
7916 0000F74B EB06                <1>       jmp    short sysvideo_15_197_2
7917                              <1> sysvideo_15_197_1:
7918 0000F74D 81C6[24580100]      <1>       add    esi, vgafont8
7919                              <1> sysvideo_15_197_2:
7920 0000F753 8B15[7E120300]      <1>       mov    edx, [maskcolor]
7921                              <1>
7922 0000F759 B108                <1>       mov    cl, 8  ; 8 rows (8x8 font)
7923                              <1>
7924 0000F75B 80FF08              <1>       cmp    bh, 8  ; 8bpp ?
7925 0000F75E 773B                <1>       ja     short sysvideo_15_197_22 ; no
7926                              <1>
7927 0000F760 3C08                <1>       cmp    al, 8  ; 8 pixels per row ?
7928 0000F762 7711                <1>       ja     short sysvideo_15_197_4   ; no
7929                              <1>
7930                              <1>       ; 8 pixels per row
7931                              <1> ;sysvideo_15_197_0:
7932                              <1> ;     lodsb
7933                              <1> ;     push   edi
7934                              <1> ;     mov    ah, 8
7935                              <1> ;sysvideo_15_197_1:
7936                              <1> ;     shr    al, 1
```

```
                            <1> ;       jnc     short sysvideo_15_197_2
                            <1> ;       mov     [edi], dl
                            <1> ;               ; character pixel/forecolor,
                            <1> ;               ; 8 bpp (256 colors)
                            <1> ;sysvideo_15_197_2:
                            <1> ;       inc     edi
                            <1> ;       dec     ah
                            <1> ;       jnz     short sysvideo_15_197_1 ; next pixel of (same) row
                            <1> ;       pop     edi
                            <1> ;       add     edi, ebp ; next row (same column position)
                            <1> ;       dec     cl
                            <1> ;       jnz     short sysvideo_15_197_0
                            <1> ;                               ; next row of the character font
0000F764 E808010000         <1>         call    sysvideo_15_197_0
                            <1>
                            <1> sysvideo_15_197_3:
                            <1>         ; 1*8*1 = 8 bytes (64 pixels)
0000F769 C605[64030300]08   <1>         mov     byte [u.r0], 8 ; font size = 8 bytes
0000F770 E977E1FFFF         <1>         jmp     sysret
                            <1>
                            <1> sysvideo_15_197_4:
0000F775 3C10               <1>         cmp     al, 16
0000F777 770A               <1>         ja      short sysvideo_15_197_10 ; >= 32 pixels per row
                            <1>         ; 16 pixels per row
                            <1> ;sysvideo_15_197_4_0:
                            <1> ;       mov     dh, dl
                            <1> ;sysvideo_15_197_5:
                            <1> ;       mov     bl, 2  ; 2 rows with same pixels
                            <1> ;sysvideo_15_197_6:
                            <1> ;       mov     al, [esi]
                            <1> ;       push    edi
                            <1> ;       mov     ah, 8
                            <1> ;sysvideo_15_197_7:
                            <1> ;       shr     al, 1
                            <1> ;       jnc     short sysvideo_15_197_8
                            <1> ;       ; 2 pixels with same forecolor
                            <1> ;       mov     [edi], dx
                            <1> ;               ; character pixel/forecolor,
                            <1> ;               ; 8 bpp (256 colors)
                            <1> ;sysvideo_15_197_8:
                            <1> ;       inc     edi
                            <1> ;       inc     edi
                            <1> ;       dec     ah
                            <1> ;       jnz     short sysvideo_15_197_7 ; next pixel of (same) row
                            <1> ;       pop     edi
                            <1> ;       add     edi, ebp ; next row
                            <1> ;       dec     bl
                            <1> ;       jnz     short sysvideo_15_197_6 ; same pixels for next row
                            <1> ;       inc     esi
                            <1> ;       dec     cl
                            <1> ;       jnz     short sysvideo_15_197_5 ; next row of the character font
0000F779 E80A010000         <1>         call    sysvideo_15_197_4_0
                            <1>
                            <1> sysvideo_15_197_9:
                            <1>         ; 2*8*2 = 32 bytes (256 pixels)
                            <1>         ;mov   byte [u.r0], 32 ; font size = 32 bytes
                            <1>         ;jmp   sysret
                            <1>         ; 03/01/2021
0000F77E E9D4FEFFFF         <1>         jmp     sysvideo_15_196_26
                            <1>
                            <1> sysvideo_15_197_10:
0000F783 3C20               <1>         cmp     al, 32
0000F785 770A               <1>         ja      short sysvideo_15_197_16 ; 64 pixels per row
                            <1>         ; 32 pixels per row
                            <1> ;sysvideo_15_197_10_0:
                            <1> ;       mov     dh, dl
                            <1> ;       ror     edx, 8  ; byte 2 -> dh, dl -> byte 3
                            <1> ;                       ; dh -> dl
                            <1> ;       mov     dh, dl ; byte 1 -> dl
                            <1> ;sysvideo_15_197_11:
                            <1> ;       mov     bl, 4  ; 4 rows with same pixels
                            <1> ;sysvideo_15_197_12:
                            <1> ;       mov     al, [esi]
                            <1> ;       push    edi
                            <1> ;       mov     ah, 8
                            <1> ;sysvideo_15_197_13:
                            <1> ;       shr     al, 1
                            <1> ;       jnc     short sysvideo_15_197_14
                            <1> ;       ; 4 pixels with same forecolor
                            <1> ;       mov     [edi], edx
                            <1> ;               ; character pixel/forecolor,
                            <1> ;               ; 8 bpp (256 colors)
                            <1> ;sysvideo_15_197_14:
                            <1> ;       add     edi, 4
                            <1> ;       dec     ah
                            <1> ;       jnz     short sysvideo_15_197_13 ; next pixel of (same) row
                            <1> ;       pop     edi
                            <1> ;       add     edi, ebp ; next row
                            <1> ;       dec     bl
                            <1> ;       jnz     short sysvideo_15_197_12 ; same pixels for next row
                            <1> ;       inc     esi
                            <1> ;       dec     cl
                            <1> ;       jnz     short sysvideo_15_197_11 ; next row of the character font
0000F787 E81F010000         <1>         call    sysvideo_15_197_10_0
                            <1>
                            <1> sysvideo_15_197_15:
                            <1>         ; 4*8*4 = 128 bytes (1024 pixels)
                            <1>         ;mov   byte [u.r0], 128 ; font size = 128 bytes
                            <1>         ;jmp   sysret
                            <1>         ; 03/01/2021
0000F78C E9DBFEFFFF         <1>         jmp     sysvideo_15_196_32
                            <1>
                            <1> sysvideo_15_197_16:
                            <1> ;       ; 64 pixels per row
                            <1> ;       mov     dh, dl
```

```
8042                            <1> ;       ror     edx, 8  ; byte 2 -> dh, dl -> byte 3
8043                            <1> ;                       ; dh -> dl
8044                            <1> ;       mov     dh, dl ; byte 1 -> dl
8045                            <1> ;sysvideo_15_197_17:
8046                            <1> ;       mov     bl, 8  ; 8 rows with same pixels
8047                            <1> ;sysvideo_15_197_18:
8048                            <1> ;       mov     al, [esi]
8049                            <1> ;       push    edi
8050                            <1> ;       mov     ah, 8
8051                            <1> ;sysvideo_15_197_19:
8052                            <1> ;       shr     al, 1
8053                            <1> ;       jnc     short sysvideo_15_197_20
8054                            <1> ;       ; 8 pixels with same forecolor
8055                            <1> ;       mov     [edi], edx
8056                            <1> ;               ; character pixel/forecolor,
8057                            <1> ;               ; 8 bpp (256 colors)
8058                            <1> ;       mov     [edi+4], edx
8059                            <1> ;               ; character pixel/forecolor,
8060                            <1> ;               ; 8 bpp (256 colors)
8061                            <1> ;sysvideo_15_197_20:
8062                            <1> ;       add     edi, 8
8063                            <1> ;       dec     ah
8064                            <1> ;       jnz     short sysvideo_15_197_19 ; next pixel of (same) row
8065                            <1> ;       pop     edi
8066                            <1> ;       add     edi, ebp ; next row
8067                            <1> ;       dec     bl
8068                            <1> ;       jnz     short sysvideo_15_197_18 ; same pixels for next row
8069                            <1> ;       inc     esi
8070                            <1> ;       dec     cl
8071                            <1> ;       jnz     short sysvideo_15_197_17 ; next row of the character font
8072 0000F791 E83D010000          <1>       call    sysvideo_15_197_16_0
8073                            <1>
8074                            <1> sysvideo_15_197_21:
8075                            <1>       ; 8*8*8 = 512 bytes (4096 pixels)
8076                            <1>       ;mov   word [u.r0], 512 ; font size = 512 bytes
8077                            <1>       ;jmp   sysret
8078                            <1>       ; 03/01/2021
8079 0000F796 E9E6FEFFFF          <1>       jmp    sysvideo_15_196_38
8080                            <1>
8081                            <1> sysvideo_15_197_22:
8082 0000F79B 80FF10              <1>       cmp    bh, 16 ; 16bpp ?
8083 0000F79E 7734                <1>       ja     short sysvideo_15_197_45 ; no
8084                            <1>
8085 0000F7A0 3C08                <1>       cmp    al, 8  ; 8 pixels per row ?
8086 0000F7A2 770A                <1>       ja     short sysvideo_15_197_27 ; no
8087                            <1>       ; 8 pixels per row
8088                            <1> ;sysvideo_15_197_23:
8089                            <1> ;       lodsb
8090                            <1> ;       push   edi
8091                            <1> ;       mov    ah, 8
8092                            <1> ;sysvideo_15_197_24:
8093                            <1> ;       shr    al, 1
8094                            <1> ;       jnc    short sysvideo_15_197_25
8095                            <1> ;       mov    [edi], dx
8096                            <1> ;              ; character pixel/forecolor,
8097                            <1> ;              ; 16 bpp (65536 colors)
8098                            <1> ;sysvideo_15_197_25:
8099                            <1> ;       inc    edi
8100                            <1> ;       inc    edi
8101                            <1> ;       dec    ah
8102                            <1> ;       jnz    short sysvideo_15_197_24 ; next pixel of (same) row
8103                            <1> ;       pop    edi
8104                            <1> ;       add    edi, ebp ; next row (same column position)
8105                            <1> ;       dec    cl
8106                            <1> ;       jnz    short sysvideo_15_197_23
8107                            <1> ;                         ; next row of the character font
8108 0000F7A4 E855010000          <1>       call   sysvideo_15_197_23
8109                            <1>
8110                            <1> sysvideo_15_197_26:
8111                            <1>       ; 1*8*1*2 = 16 bytes (64 pixels)
8112                            <1>       ;mov   byte [u.r0], 16 ; font size = 16 bytes
8113                            <1>       ;jmp   sysret
8114                            <1>       ; 03/01/2021
8115 0000F7A9 E950FEFFFF          <1>       jmp    sysvideo_15_196_3
8116                            <1>
8117                            <1> sysvideo_15_197_27:
8118 0000F7AE 3C10                <1>       cmp    al, 16
8119 0000F7B0 770A                <1>       ja     short sysvideo_15_197_33 ; >= 32 pixels per row
8120                            <1> ;sysvideo_15_197_27_0:
8121                            <1> ;       ; 16 pixels per row
8122                            <1> ;       ; (4 bytes per row)
8123                            <1> ;       mov    bx, dx
8124                            <1> ;       shl    edx, 16
8125                            <1> ;       mov    dx, bx
8126                            <1> ;sysvideo_15_197_28:
8127                            <1> ;       mov    bl, 2  ; 2 rows with same pixels
8128                            <1> ;sysvideo_15_197_29:
8129                            <1> ;       mov    al, [esi]
8130                            <1> ;       push   edi
8131                            <1> ;       mov    ah, 8
8132                            <1> ;sysvideo_15_197_30:
8133                            <1> ;       shr    al, 1
8134                            <1> ;       jnc    short sysvideo_15_197_31
8135                            <1> ;       ; 2 pixels with same forecolor
8136                            <1> ;       mov    [edi], edx
8137                            <1> ;              ; character pixel/forecolor,
8138                            <1> ;              ; 16 bpp (65536 colors)
8139                            <1> ;sysvideo_15_197_31:
8140                            <1> ;       add    edi, 4
8141                            <1> ;       dec    ah
8142                            <1> ;       jnz    short sysvideo_15_197_30 ; next pixel of (same) row
8143                            <1> ;       pop    edi
8144                            <1> ;       add    edi, ebp ; next row
8145                            <1> ;       dec    bl
8146                            <1> ;       jnz    short sysvideo_15_197_29 ; same pixels for next row
```

```
                            <1> ;       inc     esi
                            <1> ;       dec     cl
                            <1> ;       jnz     short sysvideo_15_197_28 ; next row of the character font
0000F7B2 E860010000         <1>         call    sysvideo_15_197_27_0
                            <1>
                            <1> sysvideo_15_197_32:
                            <1>         ; 2*8*2*2 = 64 bytes (256 pixels)
                            <1>         ;mov   byte [u.r0], 64 ; font size = 64 bytes
                            <1>         ;jmp   sysret
                            <1>         ; 03/01/2021
0000F7B7 E957FEFFFF         <1>         jmp     sysvideo_15_196_9
                            <1>
                            <1> sysvideo_15_197_33:
0000F7BC 3C20               <1>         cmp     al, 32
0000F7BE 770A               <1>         ja      short sysvideo_15_197_39 ; 64 pixels per row
                            <1> ;sysvideo_15_197_33_0:
                            <1> ;       ; 32 pixels per row
                            <1> ;       ; (8 bytes per row)
                            <1> ;       mov     bx, dx
                            <1> ;       shl     edx, 16
                            <1> ;       mov     dx, bx
                            <1> ;sysvideo_15_197_34:
                            <1> ;       mov     bl, 4  ; 4 rows with same pixels
                            <1> ;sysvideo_15_197_35:
                            <1> ;       mov     al, [esi]
                            <1> ;       push    edi
                            <1> ;       mov     ah, 8
                            <1> ;sysvideo_15_197_36:
                            <1> ;       shr     al, 1
                            <1> ;       jnc     short sysvideo_15_197_37
                            <1> ;       ; 4 pixels with same forecolor
                            <1> ;       mov     [edi], edx
                            <1> ;               ; character pixel/forecolor,
                            <1> ;               ; 16 bpp (65536 colors)
                            <1> ;       mov     [edi+4], edx
                            <1> ;               ; character pixel/forecolor,
                            <1> ;               ; 16 bpp (65536 colors)
                            <1> ;sysvideo_15_197_37:
                            <1> ;       add     edi, 8
                            <1> ;       dec     ah
                            <1> ;       jnz     short sysvideo_15_197_36 ; next pixel of (same) row
                            <1> ;       pop     edi
                            <1> ;       add     edi, ebp ; next row
                            <1> ;       dec     bl
                            <1> ;       jnz     short sysvideo_15_197_35 ; same pixels for next row
                            <1> ;       inc     esi
                            <1> ;       dec     cl
                            <1> ;       jnz     short sysvideo_15_197_34 ; next row of the character font
0000F7C0 E87C010000         <1>         call    sysvideo_15_197_33_0
                            <1>
                            <1> sysvideo_15_197_38:
                            <1>         ; 4*8*4*2 = 256 bytes (1024 pixels)
                            <1>         ;mov   word [u.r0], 256 ; font size = 256 bytes
                            <1>         ;jmp   sysret
                            <1>         ; 03/01/2021
0000F7C5 E95EFEFFFF         <1>         jmp     sysvideo_15_196_15
                            <1>
                            <1> sysvideo_15_197_39:
                            <1> ;       ; 64 pixels per row
                            <1> ;       ; (16 bytes per row)
                            <1> ;       mov     bx, dx
                            <1> ;       shl     edx, 16
                            <1> ;       mov     dx, bx
                            <1> ;sysvideo_15_197_40:
                            <1> ;       mov     bl, 8  ; 8 rows with same pixels
                            <1> ;sysvideo_15_197_41:
                            <1> ;       mov     al, [esi]
                            <1> ;       push    edi
                            <1> ;       mov     ah, 8
                            <1> ;sysvideo_15_197_42:
                            <1> ;       shr     al, 1
                            <1> ;       jnc     short sysvideo_15_197_43
                            <1> ;       ; 8 pixels with same forecolor
                            <1> ;       mov     [edi], edx
                            <1> ;       mov     [edi+4], edx
                            <1> ;       mov     [edi+8], edx
                            <1> ;       mov     [edi+12], edx
                            <1> ;               ; character pixel/forecolor,
                            <1> ;               ; 16 bpp (65536 colors)
                            <1> ;sysvideo_15_197_43:
                            <1> ;       add     edi, 16
                            <1> ;       dec     ah
                            <1> ;       jnz     short sysvideo_15_197_42 ; next pixel of (same) row
                            <1> ;       pop     edi
                            <1> ;       add     edi, ebp ; next row
                            <1> ;       dec     bl
                            <1> ;       jnz     short sysvideo_15_197_41 ; same pixels for next row
                            <1> ;       inc     esi
                            <1> ;       dec     cl
                            <1> ;       jnz     short sysvideo_15_197_40 ; next row of the character font
0000F7CA E89F010000         <1>         call    sysvideo_15_197_39_0
                            <1>
                            <1> sysvideo_15_197_44:
                            <1>         ; 8*8*8*2 = 1024 bytes (4096 pixels)
                            <1>         ;mov   word [u.r0], 1024 ; font size = 1024 bytes
                            <1>         ;jmp   sysret
                            <1>         ; 03/01/2021
0000F7CF E967FEFFFF         <1>         jmp     sysvideo_15_196_21
                            <1>
                            <1> sysvideo_15_197_45:
0000F7D4 80FF20             <1>         cmp     bh, 32 ; 16bpp ?
0000F7D7 7234               <1>         jb      short sysvideo_15_197_68 ; no
                            <1>
0000F7D9 3C08               <1>         cmp     al, 8  ; 8 pixels per row ?
0000F7DB 770A               <1>         ja      short sysvideo_15_197_50 ; no
```

```
8252                            <1>       ; 8 pixels per row
8253                            <1> ;sysvideo_15_197_46:
8254                            <1> ;     lodsb
8255                            <1> ;     push   edi
8256                            <1> ;     mov    ah, 8
8257                            <1> ;sysvideo_15_197_47:
8258                            <1> ;     shr    al, 1
8259                            <1> ;     jnc    short sysvideo_15_197_48
8260                            <1> ;     mov    [edi], edx
8261                            <1> ;            ; character pixel/forecolor,
8262                            <1> ;            ; 32 bpp (true colors)
8263                            <1> ;sysvideo_15_197_48:
8264                            <1> ;     add    edi, 4
8265                            <1> ;     dec    ah
8266                            <1> ;     jnz    short sysvideo_15_197_47 ; next pixel of (same) row
8267                            <1> ;     pop    edi
8268                            <1> ;     add    edi, ebp ; next row (same column position)
8269                            <1> ;     dec    cl
8270                            <1> ;     jnz    short sysvideo_15_197_46
8271                            <1> ;                           ; next row of the character font
8272 0000F7DD E8BF010000          <1>       call   sysvideo_15_197_46
8273                            <1>
8274                            <1> sysvideo_15_197_49:
8275                            <1>       ; 1*8*1*4 = 32 bytes (128 pixels)
8276                            <1>       ;mov   byte [u.r0], 32 ; font size = 32 bytes
8277                            <1>       ;jmp   sysret
8278                            <1>       ; 03/01/2021
8279 0000F7E2 E970FEFFFF          <1>       jmp    sysvideo_15_196_26
8280                            <1>
8281                            <1> sysvideo_15_197_50:
8282 0000F7E7 3C10                <1>       cmp    al, 16
8283 0000F7E9 770A                <1>       ja     short sysvideo_15_197_56 ; >= 32 pixels per row
8284                            <1> ;sysvideo_15_197_50_0:
8285                            <1> ;     ; 16 pixels per row
8286                            <1> ;     ; (8 bytes per row)
8287                            <1> ;sysvideo_15_197_51:
8288                            <1> ;     mov    bl, 2  ; 2 rows with same pixels
8289                            <1> ;sysvideo_15_197_52:
8290                            <1> ;     mov    al, [esi]
8291                            <1> ;     push   edi
8292                            <1> ;     mov    ah, 8
8293                            <1> ;sysvideo_15_197_53:
8294                            <1> ;     shr    al, 1
8295                            <1> ;     jnc    short sysvideo_15_197_54
8296                            <1> ;     ; 2 pixels with same forecolor
8297                            <1> ;     mov    [edi], edx
8298                            <1> ;     mov    [edi+4], edx
8299                            <1> ;            ; character pixel/forecolor,
8300                            <1> ;            ; 32 bpp (true colors)
8301                            <1> ;sysvideo_15_197_54:
8302                            <1> ;     add    edi, 8
8303                            <1> ;     dec    ah
8304                            <1> ;     jnz    short sysvideo_15_197_53 ; next pixel of (same) row
8305                            <1> ;     pop    edi
8306                            <1> ;     add    edi, ebp ; next row
8307                            <1> ;     dec    bl
8308                            <1> ;     jnz    short sysvideo_15_197_52 ; same pixels for next row
8309                            <1> ;     inc    esi
8310                            <1> ;     dec    cl
8311                            <1> ;     jnz    short sysvideo_15_197_51 ; next row of the character font
8312 0000F7EB E8CA010000          <1>       call   sysvideo_15_197_50_0
8313                            <1>
8314                            <1> sysvideo_15_197_55:
8315                            <1>       ; 2*8*2*4 = 128 bytes (256 pixels)
8316                            <1>       ;mov   byte [u.r0], 128 ; font size = 128 bytes
8317                            <1>       ;jmp   sysret
8318                            <1>       ; 03/01/2021
8319 0000F7F0 E977FEFFFF          <1>       jmp    sysvideo_15_196_32
8320                            <1>
8321                            <1> sysvideo_15_197_56:
8322 0000F7F5 3C20                <1>       cmp    al, 32
8323 0000F7F7 770A                <1>       ja     short sysvideo_15_197_62 ; 64 pixels per row
8324                            <1> ;sysvideo_15_197_56_0:
8325                            <1> ;     ; 32 pixels per row
8326                            <1> ;     ; (16 bytes per row)
8327                            <1> ;sysvideo_15_197_57:
8328                            <1> ;     mov    bl, 4  ; 4 rows with same pixels
8329                            <1> ;sysvideo_15_197_58:
8330                            <1> ;     mov    al, [esi]
8331                            <1> ;     push   edi
8332                            <1> ;     mov    ah, 8
8333                            <1> ;sysvideo_15_197_59:
8334                            <1> ;     shr    al, 1
8335                            <1> ;     jnc    short sysvideo_15_197_60
8336                            <1> ;     ; 4 pixels with same forecolor
8337                            <1> ;     mov    [edi], edx
8338                            <1> ;     mov    [edi+4], edx
8339                            <1> ;     mov    [edi+8], edx
8340                            <1> ;     mov    [edi+12], edx
8341                            <1> ;            ; character pixel/forecolor,
8342                            <1> ;            ; 32 bpp (true colors)
8343                            <1> ;sysvideo_15_197_60:
8344                            <1> ;     add    edi, 16
8345                            <1> ;     dec    ah
8346                            <1> ;     jnz    short sysvideo_15_197_59 ; next pixel of (same) row
8347                            <1> ;     pop    edi
8348                            <1> ;     add    edi, ebp ; next row
8349                            <1> ;     dec    bl
8350                            <1> ;     jnz    short sysvideo_15_197_58 ; same pixels for next row
8351                            <1> ;     inc    esi
8352                            <1> ;     dec    cl
8353                            <1> ;     jnz    short sysvideo_15_197_57 ; next row of the character font
8354 0000F7F9 E8E0010000          <1>       call   sysvideo_15_197_56_0
8355                            <1>
8356                            <1> sysvideo_15_197_61:
```

```
8357                                    <1>       ; 4*8*4*4 = 512 bytes (1024 pixels)
8358                                    <1>       ;mov   word [u.r0], 512 ; font size = 512 bytes
8359                                    <1>       ;jmp   sysret
8360                                    <1>       ; 03/01/2021
8361 0000F7FE E97EFEFFFF               <1>       jmp    sysvideo_15_196_38
8362                                    <1>
8363                                    <1> sysvideo_15_197_62:
8364                                    <1> ;     ; 64 pixels per row
8365                                    <1> ;     ; (16 bytes per row)
8366                                    <1> ;     mov    bx, dx
8367                                    <1> ;     shl    edx, 16
8368                                    <1> ;     mov    dx, bx
8369                                    <1> ;sysvideo_15_197_63:
8370                                    <1> ;     mov    bl, 8  ; 8 rows with same pixels
8371                                    <1> ;sysvideo_15_197_64:
8372                                    <1> ;     mov    al, [esi]
8373                                    <1> ;     push   edi
8374                                    <1> ;     mov    ah, 8
8375                                    <1> ;sysvideo_15_197_65:
8376                                    <1> ;     shr    al, 1
8377                                    <1> ;     jnc    short sysvideo_15_197_66
8378                                    <1> ;     ; 8 pixels with same forecolor
8379                                    <1> ;     mov    [edi], edx
8380                                    <1> ;     mov    [edi+4], edx
8381                                    <1> ;     mov    [edi+8], edx
8382                                    <1> ;     mov    [edi+12], edx
8383                                    <1> ;     mov    [edi+16], edx
8384                                    <1> ;     mov    [edi+20], edx
8385                                    <1> ;     mov    [edi+24], edx
8386                                    <1> ;     mov    [edi+28], edx
8387                                    <1> ;            ; character pixel/forecolor,
8388                                    <1> ;            ; 32 bpp (true colors)
8389                                    <1> ;sysvideo_15_197_66:
8390                                    <1> ;     add    edi, 32
8391                                    <1> ;     dec    ah
8392                                    <1> ;     jnz    short sysvideo_15_197_65 ; next pixel of (same) row
8393                                    <1> ;     pop    edi
8394                                    <1> ;     add    edi, ebp ; next row
8395                                    <1> ;     dec    bl
8396                                    <1> ;     jnz    short sysvideo_15_197_64 ; same pixels for next row
8397                                    <1> ;     inc    esi
8398                                    <1> ;     dec    cl
8399                                    <1> ;     jnz    short sysvideo_15_197_63 ; next row of the character font
8400 0000F803 E800020000               <1>       call   sysvideo_15_197_62_0
8401                                    <1>
8402                                    <1> sysvideo_15_197_67:
8403                                    <1>       ;; 8*8*8*4 = 2048 bytes (4096 pixels)
8404                                    <1>       ;mov   word [u.r0], 2048 ; font size = 2048 bytes
8405                                    <1>       ;jmp   sysret
8406                                    <1>       ; 03/01/2021
8407 0000F808 E987FEFFFF               <1>       jmp    sysvideo_15_196_44
8408                                    <1>
8409                                    <1> sysvideo_15_197_68:
8410                                    <1>       ; bh = 24
8411                                    <1>       ; 24 bpp
8412 0000F80D 3C08                     <1>       cmp    al, 8  ; 8 pixels per row ?
8413 0000F80F 7711                     <1>       ja     short sysvideo_15_197_73 ; no
8414                                    <1>       ; 8 pixels per row
8415                                    <1> ;sysvideo_15_197_69:
8416                                    <1> ;     lodsb
8417                                    <1> ;     push   edi
8418                                    <1> ;     mov    ah, 8
8419                                    <1> ;sysvideo_15_197_70:
8420                                    <1> ;     shr    al, 1
8421                                    <1> ;     jnc    short sysvideo_15_197_71
8422                                    <1> ;     ;mov   ebx, edx
8423                                    <1> ;     ;shr   ebx, 16
8424                                    <1> ;     ;mov   [edi], dx
8425                                    <1> ;     ;mov   [edi+2], bl
8426                                    <1> ;     call   sysvideo_15_196_91
8427                                    <1> ;            ; character pixel/forecolor,
8428                                    <1> ;            ; 24 bpp (true colors)
8429                                    <1> ;sysvideo_15_197_71:
8430                                    <1> ;     add    edi, 3
8431                                    <1> ;     dec    ah
8432                                    <1> ;     jnz    short sysvideo_15_197_70 ; next pixel of (same) row
8433                                    <1> ;     pop    edi
8434                                    <1> ;     add    edi, ebp ; next row (same column position)
8435                                    <1> ;     dec    cl
8436                                    <1> ;     jnz    short sysvideo_15_197_69
8437                                    <1> ;                           ; next row of the character font
8438 0000F811 E831020000               <1>       call   sysvideo_15_197_69
8439                                    <1>
8440                                    <1> sysvideo_15_197_72:
8441                                    <1>       ; 1*8*1*3 = 24 bytes (64 pixels)
8442 0000F816 C605[64030300]18         <1>       mov    byte [u.r0], 24 ; font size = 24 bytes
8443 0000F81D E9CAE0FFFF               <1>       jmp    sysret
8444                                    <1>
8445                                    <1> sysvideo_15_197_73:
8446 0000F822 3C10                     <1>       cmp    al, 16
8447 0000F824 7711                     <1>       ja     short sysvideo_15_197_79 ; >= 32 pixels per row
8448                                    <1>       ; 16 pixels per row
8449                                    <1>       ; (6 bytes per row)
8450                                    <1> ;sysvideo_15_197_74:
8451                                    <1> ;     mov    bl, 2  ; 2 rows with same pixels
8452                                    <1> ;sysvideo_15_197_75:
8453                                    <1> ;     mov    al, [esi]
8454                                    <1> ;     push   edi
8455                                    <1> ;     mov    ah, 8
8456                                    <1> ;sysvideo_15_197_76:
8457                                    <1> ;     shr    al, 1
8458                                    <1> ;     jnc    short sysvideo_15_197_77
8459                                    <1> ;     ; 2 pixels with same forecolor
8460                                    <1> ;     ;mov   ebx, edx
8461                                    <1> ;     ;shr   ebx, 16
```

```
                            <1> ;       ;mov    [edi], dx
                            <1> ;       ;mov    [edi+2], bl
                            <1> ;       call    sysvideo_15_196_91
                            <1> ;       ;mov    [edi+3], dx
                            <1> ;       ;mov    [edi+5], bl
                            <1> ;       call    sysvideo_15_196_92
                            <1> ;               ; character pixel/forecolor,
                            <1> ;               ; 24 bpp (true colors)
                            <1> ;sysvideo_15_197_77:
                            <1> ;       add     edi, 6
                            <1> ;       dec     ah
                            <1> ;       jnz     short sysvideo_15_197_76 ; next pixel of (same) row
                            <1> ;       pop     edi
                            <1> ;       add     edi, ebp ; next row
                            <1> ;       dec     bl
                            <1> ;       jnz     short sysvideo_15_197_75 ; same pixels for next row
                            <1> ;       inc     esi
                            <1> ;       dec     cl
                            <1> ;       jnz     short sysvideo_15_197_74 ; next row of the character font
0000F826 E838020000         <1>         call    sysvideo_15_197_74
                            <1>
                            <1> sysvideo_15_197_78:
                            <1>         ; 2*8*2*3 = 96 bytes (256 pixels)
0000F82B C605[64030300]60   <1>         mov     byte [u.r0], 96 ; font size = 96 bytes
0000F832 E9B5E0FFFF         <1>         jmp     sysret
                            <1>
                            <1> sysvideo_15_197_79:
0000F837 3C20               <1>         cmp     al, 32
0000F839 7713               <1>         ja      short sysvideo_15_197_85 ; 64 pixels per row
                            <1>         ; 32 pixels per row
                            <1>         ; (12 bytes per row)
                            <1> ;sysvideo_15_197_80:
                            <1> ;       mov     bl, 4  ; 4 rows with same pixels
                            <1> ;sysvideo_15_197_81:
                            <1> ;       mov     al, [esi]
                            <1> ;       push    edi
                            <1> ;       mov     ah, 8
                            <1> ;sysvideo_15_197_82:
                            <1> ;       shr     al, 1
                            <1> ;       jnc     short sysvideo_15_197_83
                            <1> ;       ; 4 pixels with same forecolor
                            <1> ;       ;mov    ebx, edx
                            <1> ;       ;shr    ebx, 16
                            <1> ;       ;mov    [edi], dx
                            <1> ;       ;mov    [edi+2], bl
                            <1> ;       call    sysvideo_15_196_91
                            <1> ;       ;mov    [edi+3], dx
                            <1> ;       ;mov    [edi+5], bl
                            <1> ;       ;mov    [edi+6], dx
                            <1> ;       ;mov    [edi+8], bl
                            <1> ;       ;mov    [edi+9], dx
                            <1> ;       ;mov    [edi+11], bl
                            <1> ;       call    sysvideo_15_196_93
                            <1> ;               ; character pixel/forecolor,
                            <1> ;               ; 24 bpp (true colors)
                            <1> ;sysvideo_15_197_83:
                            <1> ;       add     edi, 12
                            <1> ;       dec     ah
                            <1> ;       jnz     short sysvideo_15_197_82 ; next pixel of (same) row
                            <1> ;       pop     edi
                            <1> ;       add     edi, ebp ; next row
                            <1> ;       dec     bl
                            <1> ;       jnz     short sysvideo_15_197_81 ; same pixels for next row
                            <1> ;       inc     esi
                            <1> ;       dec     cl
                            <1> ;       jnz     short sysvideo_15_197_80 ; next row of the character font
0000F83B E84C020000         <1>         call    sysvideo_15_197_80
                            <1>
                            <1> sysvideo_15_197_84:
                            <1>         ; 4*8*4*3 = 384 bytes (1024 pixels)
0000F840 66C705[64030300]80- <1>        mov     word [u.r0], 384 ; font size = 384 bytes
0000F848 01                 <1>
0000F849 E99EE0FFFF         <1>         jmp     sysret
                            <1>
                            <1> sysvideo_15_197_85:
                            <1> ;       ; 64 pixels per row
                            <1> ;       ; (24 bytes per row)
                            <1> ;       mov     bx, dx
                            <1> ;       shl     edx, 16
                            <1> ;       mov     dx, bx
                            <1> ;sysvideo_15_197_86:
                            <1> ;       mov     bl, 8  ; 8 rows with same pixels
                            <1> ;sysvideo_15_197_87:
                            <1> ;       mov     al, [esi]
                            <1> ;       push    edi
                            <1> ;       mov     ah, 8
                            <1> ;sysvideo_15_197_88:
                            <1> ;       shr     al, 1
                            <1> ;       jnc     short sysvideo_15_197_89
                            <1> ;       ; 8 pixels with same forecolor
                            <1> ;       ;mov    ebx, edx
                            <1> ;       ;shr    ebx, 16
                            <1> ;       ;mov    [edi], dx
                            <1> ;       ;mov    [edi+2], bl
                            <1> ;       call    sysvideo_15_196_91
                            <1> ;       ;mov    [edi+3], dx
                            <1> ;       ;mov    [edi+5], bl
                            <1> ;       ;mov    [edi+6], dx
                            <1> ;       ;mov    [edi+8], bl
                            <1> ;       ;mov    [edi+9], dx
                            <1> ;       ;mov    [edi+11], bl
                            <1> ;       ;mov    [edi+12], dx
                            <1> ;       ;mov    [edi+14], bl
                            <1> ;       ;mov    [edi+15], dx
                            <1> ;       ;mov    [edi+17], bl
```

```
8566                                    <1> ;       ;mov   [edi+18], dx
8567                                    <1> ;       ;mov   [edi+20], bl
8568                                    <1> ;       ;mov   [edi+21], dx
8569                                    <1> ;       ;mov   [edi+23], bl
8570                                    <1> ;       call   sysvideo_15_196_94
8571                                    <1> ;              ; character pixel/forecolor,
8572                                    <1> ;              ; 24 bpp (true colors)
8573                                    <1> ;sysvideo_15_197_89:
8574                                    <1> ;       add    edi, 24
8575                                    <1> ;       dec    ah
8576                                    <1> ;       jnz    short sysvideo_15_197_88 ; next pixel of (same) row
8577                                    <1> ;       pop    edi
8578                                    <1> ;       add    edi, ebp ; next row
8579                                    <1> ;       dec    bl
8580                                    <1> ;       jnz    short sysvideo_15_197_87 ; same pixels for next row
8581                                    <1> ;       inc    esi
8582                                    <1> ;       dec    cl
8583                                    <1> ;       jnz    short sysvideo_15_197_86 ; next row of the character font
8584 0000F84E E862020000               <1>         call   sysvideo_15_197_85_0
8585                                    <1>
8586                                    <1> sysvideo_15_197_90:
8587                                    <1>         ; 8*8*8*3 = 1536 bytes (4096 pixels)
8588 0000F853 66C705[64030300]00-       <1>         mov    word [u.r0], 1536 ; font size = 1536 bytes
8588 0000F85B 06                        <1>
8589 0000F85C E98BE0FFFF                <1>         jmp    sysret
8590                                    <1>
8591                                    <1>
8592                                    <1> char_size_option:
8593                                    <1>         ; 0 = 8x16, 1 = 8x8, 2 = 2x(8x16), 3 = 2x(8x8)
8594                                    <1>         ; 4 = 4x(8x16), 5 = 4x(8x8), 6 = 8x(8x16), 7 = 8x(8x8)
8595                                    <1>         ; character size (while writing at LFB address)
8596 0000F861 0810080810201010           <1>         dw     1008h, 0808h, 2010h, 1010h
8597 0000F869 2040202040804040          <1>         dw     4020h, 2020h, 8040h, 4040h
8598                                    <1>
8599                                    <1> sysvideo_15_196_0:
8600                                    <1> sysvideo_15_197_0:
8601                                    <1>         ; 03/01/2021
8602                                    <1>         ; 8 pixels per row
8603 0000F871 AC                       <1>         lodsb
8604 0000F872 57                        <1>         push   edi
8605 0000F873 B408                      <1>         mov    ah, 8
8606                                    <1> sysvideo_15_196_0_1:
8607 0000F875 D0E8                      <1>         shr    al, 1
8608 0000F877 7302                      <1>         jnc    short sysvideo_15_196_0_2
8609 0000F879 8817                      <1>         mov    [edi], dl
8610                                    <1>                ; character pixel/forecolor,
8611                                    <1>                ; 8 bpp (256 colors)
8612                                    <1> sysvideo_15_196_0_2:
8613 0000F87B 47                        <1>         inc    edi
8614 0000F87C FECC                      <1>         dec    ah
8615 0000F87E 75F5                      <1>         jnz    short sysvideo_15_196_0_1
8616                                    <1>                               ; next pixel of (same) row
8617 0000F880 5F                        <1>         pop    edi
8618 0000F881 01EF                      <1>         add    edi, ebp     ; next row (same column position)
8619 0000F883 FEC9                      <1>         dec    cl
8620 0000F885 75EA                      <1>         jnz    short sysvideo_15_196_0
8621                                    <1>                               ; next row of the character font
8622 0000F887 C3                        <1>         retn
8623                                    <1>
8624                                    <1> sysvideo_15_196_4_0:
8625                                    <1> sysvideo_15_197_4_0:
8626                                    <1>         ; 16 pixels per row
8627 0000F888 88D6                      <1>         mov    dh, dl
8628                                    <1> sysvideo_15_196_5:
8629 0000F88A B302                      <1>         mov    bl, 2  ; 2 rows with same pixels
8630                                    <1> sysvideo_15_196_6:
8631 0000F88C 8A06                      <1>         mov    al, [esi]
8632 0000F88E 57                        <1>         push   edi
8633 0000F88F B408                      <1>         mov    ah, 8
8634                                    <1> sysvideo_15_196_7:
8635 0000F891 D0E8                      <1>         shr    al, 1
8636 0000F893 7303                      <1>         jnc    short sysvideo_15_196_8
8637                                    <1>         ; 2 pixels with same forecolor
8638 0000F895 668917                    <1>         mov    [edi], dx
8639                                    <1>                ; character pixel/forecolor,
8640                                    <1>                ; 8 bpp (256 colors)
8641                                    <1> sysvideo_15_196_8:
8642 0000F898 47                        <1>         inc    edi
8643 0000F899 47                        <1>         inc    edi
8644 0000F89A FECC                      <1>         dec    ah
8645 0000F89C 75F3                      <1>         jnz    short sysvideo_15_196_7 ; next pixel of (same) row
8646 0000F89E 5F                        <1>         pop    edi
8647 0000F89F 01EF                      <1>         add    edi, ebp ; next row
8648 0000F8A1 FECB                      <1>         dec    bl
8649 0000F8A3 75E7                      <1>         jnz    short sysvideo_15_196_6 ; same pixels for next row
8650 0000F8A5 46                        <1>         inc    esi
8651 0000F8A6 FEC9                      <1>         dec    cl
8652 0000F8A8 75E0                      <1>         jnz    short sysvideo_15_196_5 ; next row of the character font
8653 0000F8AA C3                        <1>         retn
8654                                    <1>
8655                                    <1> sysvideo_15_196_10_0:
8656                                    <1> sysvideo_15_197_10_0:
8657                                    <1>         ; 32 pixels per row
8658 0000F8AB 88D6                      <1>         mov    dh, dl
8659 0000F8AD C1CA08                    <1>         ror    edx, 8  ; byte 2 -> dh, dl -> byte 3
8660                                    <1>                        ; dh -> dl
8661 0000F8B0 88D6                      <1>         mov    dh, dl ; byte 1 -> dl
8662                                    <1> sysvideo_15_196_11:
8663 0000F8B2 B304                      <1>         mov    bl, 4  ; 4 rows with same pixels
8664                                    <1> sysvideo_15_196_12:
8665 0000F8B4 8A06                      <1>         mov    al, [esi]
8666 0000F8B6 57                        <1>         push   edi
8667 0000F8B7 B408                      <1>         mov    ah, 8
8668                                    <1> sysvideo_15_196_13:
8669 0000F8B9 D0E8                      <1>         shr    al, 1
```

```
8670 0000F8BB 7302              <1>       jnc    short sysvideo_15_196_14
8671                            <1>       ; 4 pixels with same forecolor
8672 0000F8BD 8917              <1>       mov    [edi], edx
8673                            <1>              ; character pixel/forecolor,
8674                            <1>              ; 8 bpp (256 colors)
8675                            <1> sysvideo_15_196_14:
8676 0000F8BF 83C704            <1>       add    edi, 4
8677 0000F8C2 FECC              <1>       dec    ah
8678 0000F8C4 75F3              <1>       jnz    short sysvideo_15_196_13 ; next pixel of (same) row
8679 0000F8C6 5F                <1>       pop    edi
8680 0000F8C7 01EF              <1>       add    edi, ebp ; next row
8681 0000F8C9 FECB              <1>       dec    bl
8682 0000F8CB 75E7              <1>       jnz    short sysvideo_15_196_12 ; same pixels for next row
8683 0000F8CD 46                <1>       inc    esi
8684 0000F8CE FEC9              <1>       dec    cl
8685 0000F8D0 75E0              <1>       jnz    short sysvideo_15_196_11 ; next row of the character font
8686 0000F8D2 C3                <1>       retn
8687                            <1>
8688                            <1> sysvideo_15_196_16_0:
8689                            <1> sysvideo_15_197_16_0:
8690                            <1>       ; 64 pixels per row
8691 0000F8D3 88D6              <1>       mov    dh, dl
8692 0000F8D5 C1CA08            <1>       ror    edx, 8  ; byte 2 -> dh, dl -> byte 3
8693                            <1>                      ; dh -> dl
8694 0000F8D8 88D6              <1>       mov    dh, dl ; byte 1 -> dl
8695                            <1> sysvideo_15_196_17:
8696 0000F8DA B308              <1>       mov    bl, 8  ; 8 rows with same pixels
8697                            <1> sysvideo_15_196_18:
8698 0000F8DC 8A06              <1>       mov    al, [esi]
8699 0000F8DE 57                <1>       push   edi
8700 0000F8DF B408              <1>       mov    ah, 8
8701                            <1> sysvideo_15_196_19:
8702 0000F8E1 D0E8              <1>       shr    al, 1
8703 0000F8E3 7305              <1>       jnc    short sysvideo_15_196_20
8704                            <1>       ; 8 pixels with same forecolor
8705 0000F8E5 8917              <1>       mov    [edi], edx
8706                            <1>              ; character pixel/forecolor,
8707                            <1>              ; 8 bpp (256 colors)
8708 0000F8E7 895704            <1>       mov    [edi+4], edx
8709                            <1>              ; character pixel/forecolor,
8710                            <1>              ; 8 bpp (256 colors)
8711                            <1> sysvideo_15_196_20:
8712 0000F8EA 83C708            <1>       add    edi, 8
8713 0000F8ED FECC              <1>       dec    ah
8714 0000F8EF 75F0              <1>       jnz    short sysvideo_15_196_19 ; next pixel of (same) row
8715 0000F8F1 5F                <1>       pop    edi
8716 0000F8F2 01EF              <1>       add    edi, ebp ; next row
8717 0000F8F4 FECB              <1>       dec    bl
8718 0000F8F6 75E4              <1>       jnz    short sysvideo_15_196_18 ; same pixels for next row
8719 0000F8F8 46                <1>       inc    esi
8720 0000F8F9 FEC9              <1>       dec    cl
8721 0000F8FB 75DD              <1>       jnz    short sysvideo_15_196_17 ; next row of the character font
8722 0000F8FD C3                <1>       retn
8723                            <1>
8724                            <1> sysvideo_15_196_23:
8725                            <1> sysvideo_15_197_23:
8726                            <1>       ; 03/01/2021
8727                            <1>       ; 8 pixels per row
8728 0000F8FE AC                <1>       lodsb
8729 0000F8FF 57                <1>       push   edi
8730 0000F900 B408              <1>       mov    ah, 8
8731                            <1> sysvideo_15_196_24:
8732 0000F902 D0E8              <1>       shr    al, 1
8733 0000F904 7303              <1>       jnc    short sysvideo_15_196_25
8734 0000F906 668917            <1>       mov    [edi], dx
8735                            <1>              ; character pixel/forecolor,
8736                            <1>              ; 16 bpp (65536 colors)
8737                            <1> sysvideo_15_196_25:
8738 0000F909 47                <1>       inc    edi
8739 0000F90A 47                <1>       inc    edi
8740 0000F90B FECC              <1>       dec    ah
8741 0000F90D 75F3              <1>       jnz    short sysvideo_15_196_24 ; next pixel of (same) row
8742 0000F90F 5F                <1>       pop    edi
8743 0000F910 01EF              <1>       add    edi, ebp ; next row (same column position)
8744 0000F912 FEC9              <1>       dec    cl
8745 0000F914 75E8              <1>       jnz    short sysvideo_15_196_23
8746                            <1>                           ; next row of the character font
8747 0000F916 C3                <1>       retn
8748                            <1>
8749                            <1> sysvideo_15_196_27_0:
8750                            <1> sysvideo_15_197_27_0:
8751                            <1>       ; 16 pixels per row
8752                            <1>       ; (4 bytes per row)
8753 0000F917 6689D3            <1>       mov    bx, dx
8754 0000F91A C1E210            <1>       shl    edx, 16
8755 0000F91D 6689DA            <1>       mov    dx, bx
8756                            <1> sysvideo_15_196_28:
8757 0000F920 B302              <1>       mov    bl, 2  ; 2 rows with same pixels
8758                            <1> sysvideo_15_196_29:
8759 0000F922 8A06              <1>       mov    al, [esi]
8760 0000F924 57                <1>       push   edi
8761 0000F925 B408              <1>       mov    ah, 8
8762                            <1> sysvideo_15_196_30:
8763 0000F927 D0E8              <1>       shr    al, 1
8764 0000F929 7302              <1>       jnc    short sysvideo_15_196_31
8765                            <1>       ; 2 pixels with same forecolor
8766 0000F92B 8917              <1>       mov    [edi], edx
8767                            <1>              ; character pixel/forecolor,
8768                            <1>              ; 16 bpp (65536 colors)
8769                            <1> sysvideo_15_196_31:
8770 0000F92D 83C704            <1>       add    edi, 4
8771 0000F930 FECC              <1>       dec    ah
8772 0000F932 75F3              <1>       jnz    short sysvideo_15_196_30 ; next pixel of (same) row
8773 0000F934 5F                <1>       pop    edi
8774 0000F935 01EF              <1>       add    edi, ebp ; next row
```

```
8775 0000F937 FECB              <1>       dec    bl
8776 0000F939 75E7              <1>       jnz    short sysvideo_15_196_29 ; same pixels for next row
8777 0000F93B 46                <1>       inc    esi
8778 0000F93C FEC9              <1>       dec    cl
8779 0000F93E 75E0              <1>       jnz    short sysvideo_15_196_28 ; next row of the character font
8780 0000F940 C3                <1>       retn
8781                            <1>
8782                            <1> sysvideo_15_196_33_0:
8783                            <1> sysvideo_15_197_33_0:
8784                            <1>       ; 32 pixels per row
8785                            <1>       ; (8 bytes per row)
8786 0000F941 6689D3            <1>       mov    bx, dx
8787 0000F944 C1E210            <1>       shl    edx, 16
8788 0000F947 6689DA            <1>       mov    dx, bx
8789                            <1> sysvideo_15_196_34:
8790 0000F94A B304              <1>       mov    bl, 4  ; 4 rows with same pixels
8791                            <1> sysvideo_15_196_35:
8792 0000F94C 8A06              <1>       mov    al, [esi]
8793 0000F94E 57                <1>       push   edi
8794 0000F94F B408              <1>       mov    ah, 8
8795                            <1> sysvideo_15_196_36:
8796 0000F951 D0E8              <1>       shr    al, 1
8797 0000F953 7305              <1>       jnc    short sysvideo_15_196_37
8798                            <1>       ; 4 pixels with same forecolor
8799 0000F955 8917              <1>       mov    [edi], edx
8800                            <1>              ; character pixel/forecolor,
8801                            <1>              ; 16 bpp (65536 colors)
8802 0000F957 895704            <1>       mov    [edi+4], edx
8803                            <1>              ; character pixel/forecolor,
8804                            <1>              ; 16 bpp (65536 colors)
8805                            <1> sysvideo_15_196_37:
8806 0000F95A 83C708            <1>       add    edi, 8
8807 0000F95D FECC              <1>       dec    ah
8808 0000F95F 75F0              <1>       jnz    short sysvideo_15_196_36 ; next pixel of (same) row
8809 0000F961 5F                <1>       pop    edi
8810 0000F962 01EF              <1>       add    edi, ebp ; next row
8811 0000F964 FECB              <1>       dec    bl
8812 0000F966 75E4              <1>       jnz    short sysvideo_15_196_35 ; same pixels for next row
8813 0000F968 46                <1>       inc    esi
8814 0000F969 FEC9              <1>       dec    cl
8815 0000F96B 75DD              <1>       jnz    short sysvideo_15_196_34 ; next row of the character font
8816 0000F96D C3                <1>       retn
8817                            <1>
8818                            <1> sysvideo_15_196_39_0:
8819                            <1> sysvideo_15_197_39_0:
8820                            <1>       ; 64 pixels per row
8821                            <1>       ; (16 bytes per row)
8822 0000F96E 6689D3            <1>       mov    bx, dx
8823 0000F971 C1E210            <1>       shl    edx, 16
8824 0000F974 6689DA            <1>       mov    dx, bx
8825                            <1> sysvideo_15_196_40:
8826 0000F977 B308              <1>       mov    bl, 8  ; 8 rows with same pixels
8827                            <1> sysvideo_15_196_41:
8828 0000F979 8A06              <1>       mov    al, [esi]
8829 0000F97B 57                <1>       push   edi
8830 0000F97C B408              <1>       mov    ah, 8
8831                            <1> sysvideo_15_196_42:
8832 0000F97E D0E8              <1>       shr    al, 1
8833 0000F980 730B              <1>       jnc    short sysvideo_15_196_43
8834                            <1>       ; 8 pixels with same forecolor
8835 0000F982 8917              <1>       mov    [edi], edx
8836 0000F984 895704            <1>       mov    [edi+4], edx
8837 0000F987 895708            <1>       mov    [edi+8], edx
8838 0000F98A 89570C            <1>       mov    [edi+12], edx
8839                            <1>              ; character pixel/forecolor,
8840                            <1>              ; 16 bpp (65536 colors)
8841                            <1> sysvideo_15_196_43:
8842 0000F98D 83C710            <1>       add    edi, 16
8843 0000F990 FECC              <1>       dec    ah
8844 0000F992 75EA              <1>       jnz    short sysvideo_15_196_42 ; next pixel of (same) row
8845 0000F994 5F                <1>       pop    edi
8846 0000F995 01EF              <1>       add    edi, ebp ; next row
8847 0000F997 FECB              <1>       dec    bl
8848 0000F999 75DE              <1>       jnz    short sysvideo_15_196_41 ; same pixels for next row
8849 0000F99B 46                <1>       inc    esi
8850 0000F99C FEC9              <1>       dec    cl
8851 0000F99E 75D7              <1>       jnz    short sysvideo_15_196_40 ; next row of the character font
8852 0000F9A0 C3                <1>       retn
8853                            <1>
8854                            <1> sysvideo_15_196_46:
8855                            <1> sysvideo_15_197_46:
8856                            <1>       ; 8 pixels per row
8857 0000F9A1 AC                <1>       lodsb
8858 0000F9A2 57                <1>       push   edi
8859 0000F9A3 B408              <1>       mov    ah, 8
8860                            <1> sysvideo_15_196_47:
8861 0000F9A5 D0E8              <1>       shr    al, 1
8862 0000F9A7 7302              <1>       jnc    short sysvideo_15_196_48
8863 0000F9A9 8917              <1>       mov    [edi], edx
8864                            <1>              ; character pixel/forecolor,
8865                            <1>              ; 32 bpp (true colors)
8866                            <1> sysvideo_15_196_48:
8867 0000F9AB 83C704            <1>       add    edi, 4
8868 0000F9AE FECC              <1>       dec    ah
8869 0000F9B0 75F3              <1>       jnz    short sysvideo_15_196_47 ; next pixel of (same) row
8870 0000F9B2 5F                <1>       pop    edi
8871 0000F9B3 01EF              <1>       add    edi, ebp ; next row (same column position)
8872 0000F9B5 FEC9              <1>       dec    cl
8873 0000F9B7 75E8              <1>       jnz    short sysvideo_15_196_46
8874                            <1>                           ; next row of the character font
8875 0000F9B9 C3                <1>       retn
8876                            <1>
8877                            <1> sysvideo_15_196_50_0:
8878                            <1> sysvideo_15_197_50_0:
8879                            <1>       ; 16 pixels per row
```

```
8880                            <1>       ; (8 bytes per row)
8881                            <1> sysvideo_15_196_51:
8882 0000F9BA B302              <1>       mov   bl, 2  ; 2 rows with same pixels
8883                            <1> sysvideo_15_196_52:
8884 0000F9BC 8A06              <1>       mov   al, [esi]
8885 0000F9BE 57                <1>       push  edi
8886 0000F9BF B408              <1>       mov   ah, 8
8887                            <1> sysvideo_15_196_53:
8888 0000F9C1 D0E8              <1>       shr   al, 1
8889 0000F9C3 7305              <1>       jnc   short sysvideo_15_196_54
8890                            <1>       ; 2 pixels with same forecolor
8891 0000F9C5 8917              <1>       mov   [edi], edx
8892 0000F9C7 895704            <1>       mov   [edi+4], edx
8893                            <1>             ; character pixel/forecolor,
8894                            <1>             ; 32 bpp (true colors)
8895                            <1> sysvideo_15_196_54:
8896 0000F9CA 83C708            <1>       add   edi, 8
8897 0000F9CD FECC              <1>       dec   ah
8898 0000F9CF 75F0              <1>       jnz   short sysvideo_15_196_53 ; next pixel of (same) row
8899 0000F9D1 5F                <1>       pop   edi
8900 0000F9D2 01EF              <1>       add   edi, ebp ; next row
8901 0000F9D4 FECB              <1>       dec   bl
8902 0000F9D6 75E4              <1>       jnz   short sysvideo_15_196_52 ; same pixels for next row
8903 0000F9D8 46                <1>       inc   esi
8904 0000F9D9 FEC9              <1>       dec   cl
8905 0000F9DB 75DD              <1>       jnz   short sysvideo_15_196_51 ; next row of the character font
8906 0000F9DD C3                <1>       retn
8907                            <1>
8908                            <1> sysvideo_15_196_56_0:
8909                            <1> sysvideo_15_197_56_0:
8910                            <1>       ; 32 pixels per row
8911                            <1>       ; (16 bytes per row)
8912                            <1> sysvideo_15_196_57:
8913 0000F9DE B304              <1>       mov   bl, 4  ; 4 rows with same pixels
8914                            <1> sysvideo_15_196_58:
8915 0000F9E0 8A06              <1>       mov   al, [esi]
8916 0000F9E2 57                <1>       push  edi
8917 0000F9E3 B408              <1>       mov   ah, 8
8918                            <1> sysvideo_15_196_59:
8919 0000F9E5 D0E8              <1>       shr   al, 1
8920 0000F9E7 730B              <1>       jnc   short sysvideo_15_196_60
8921                            <1>       ; 4 pixels with same forecolor
8922 0000F9E9 8917              <1>       mov   [edi], edx
8923 0000F9EB 895704            <1>       mov   [edi+4], edx
8924 0000F9EE 895708            <1>       mov   [edi+8], edx
8925 0000F9F1 89570C            <1>       mov   [edi+12], edx
8926                            <1>             ; character pixel/forecolor,
8927                            <1>             ; 32 bpp (true colors)
8928                            <1> sysvideo_15_196_60:
8929 0000F9F4 83C710            <1>       add   edi, 16
8930 0000F9F7 FECC              <1>       dec   ah
8931 0000F9F9 75EA              <1>       jnz   short sysvideo_15_196_59 ; next pixel of (same) row
8932 0000F9FB 5F                <1>       pop   edi
8933 0000F9FC 01EF              <1>       add   edi, ebp ; next row
8934 0000F9FE FECB              <1>       dec   bl
8935 0000FA00 75DE              <1>       jnz   short sysvideo_15_196_58 ; same pixels for next row
8936 0000FA02 46                <1>       inc   esi
8937 0000FA03 FEC9              <1>       dec   cl
8938 0000FA05 75D7              <1>       jnz   short sysvideo_15_196_57 ; next row of the character font
8939 0000FA07 C3                <1>       retn
8940                            <1>
8941                            <1> sysvideo_15_196_62_0:
8942                            <1> sysvideo_15_197_62_0:
8943                            <1>       ; 64 pixels per row
8944                            <1>       ; (16 bytes per row)
8945 0000FA08 6689D3            <1>       mov   bx, dx
8946 0000FA0B C1E210            <1>       shl   edx, 16
8947 0000FA0E 6689DA            <1>       mov   dx, bx
8948                            <1> sysvideo_15_196_63:
8949 0000FA11 B308              <1>       mov   bl, 8  ; 8 rows with same pixels
8950                            <1> sysvideo_15_196_64:
8951 0000FA13 8A06              <1>       mov   al, [esi]
8952 0000FA15 57                <1>       push  edi
8953 0000FA16 B408              <1>       mov   ah, 8
8954                            <1> sysvideo_15_196_65:
8955 0000FA18 D0E8              <1>       shr   al, 1
8956 0000FA1A 7317              <1>       jnc   short sysvideo_15_196_66
8957                            <1>       ; 8 pixels with same forecolor
8958 0000FA1C 8917              <1>       mov   [edi], edx
8959 0000FA1E 895704            <1>       mov   [edi+4], edx
8960 0000FA21 895708            <1>       mov   [edi+8], edx
8961 0000FA24 89570C            <1>       mov   [edi+12], edx
8962 0000FA27 895710            <1>       mov   [edi+16], edx
8963 0000FA2A 895714            <1>       mov   [edi+20], edx
8964 0000FA2D 895718            <1>       mov   [edi+24], edx
8965 0000FA30 89571C            <1>       mov   [edi+28], edx
8966                            <1>             ; character pixel/forecolor,
8967                            <1>             ; 32 bpp (true colors)
8968                            <1> sysvideo_15_196_66:
8969 0000FA33 83C720            <1>       add   edi, 32
8970 0000FA36 FECC              <1>       dec   ah
8971 0000FA38 75DE              <1>       jnz   short sysvideo_15_196_65 ; next pixel of (same) row
8972 0000FA3A 5F                <1>       pop   edi
8973 0000FA3B 01EF              <1>       add   edi, ebp ; next row
8974 0000FA3D FECB              <1>       dec   bl
8975 0000FA3F 75D2              <1>       jnz   short sysvideo_15_196_64 ; same pixels for next row
8976 0000FA41 46                <1>       inc   esi
8977 0000FA42 FEC9              <1>       dec   cl
8978 0000FA44 75CB              <1>       jnz   short sysvideo_15_196_63 ; next row of the character font
8979 0000FA46 C3                <1>       retn
8980                            <1>
8981                            <1> sysvideo_15_196_69:
8982                            <1> sysvideo_15_197_69:
8983                            <1>       ; 03/01/2021
8984 0000FA47 AC                <1>       lodsb
```

```
8985 0000FA48 57                <1>       push   edi
8986 0000FA49 B408              <1>       mov    ah, 8
8987                            <1> sysvideo_15_196_70:
8988 0000FA4B D0E8              <1>       shr    al, 1
8989 0000FA4D 7305              <1>       jnc    short sysvideo_15_196_71
8990                            <1>       ;mov   ebx, edx
8991                            <1>       ;shr   ebx, 16
8992                            <1>       ;mov   [edi], dx
8993                            <1>       ;mov   [edi+2], bl
8994 0000FA4F E893000000        <1>       call   sysvideo_15_196_91
8995                            <1>              ; character pixel/forecolor,
8996                            <1>              ; 24 bpp (true colors)
8997                            <1> sysvideo_15_196_71:
8998 0000FA54 83C703            <1>       add    edi, 3
8999 0000FA57 FECC              <1>       dec    ah
9000 0000FA59 75F0              <1>       jnz    short sysvideo_15_196_70 ; next pixel of (same) row
9001 0000FA5B 5F                <1>       pop    edi
9002 0000FA5C 01EF              <1>       add    edi, ebp ; next row (same column position)
9003 0000FA5E FEC9              <1>       dec    cl
9004 0000FA60 75E5              <1>       jnz    short sysvideo_15_196_69
9005                            <1>                              ; next row of the character font
9006 0000FA62 C3                <1>       retn
9007                            <1>
9008                            <1> sysvideo_15_196_74:
9009                            <1> sysvideo_15_197_74:
9010                            <1>       ; 16 pixels per row
9011                            <1>       ; (6 bytes per row)
9012 0000FA63 B302              <1>       mov    bl, 2  ; 2 rows with same pixels
9013                            <1> sysvideo_15_196_75:
9014 0000FA65 8A06              <1>       mov    al, [esi]
9015 0000FA67 57                <1>       push   edi
9016 0000FA68 B408              <1>       mov    ah, 8
9017                            <1> sysvideo_15_196_76:
9018 0000FA6A D0E8              <1>       shr    al, 1
9019 0000FA6C 730A              <1>       jnc    short sysvideo_15_196_77
9020                            <1>       ; 2 pixels with same forecolor
9021                            <1>       ;mov   ebx, edx
9022                            <1>       ;shr   ebx, 16
9023                            <1>       ;mov   [edi], dx
9024                            <1>       ;mov   [edi+2], bl
9025 0000FA6E E874000000        <1>       call   sysvideo_15_196_91
9026                            <1>       ;mov   [edi+3], dx
9027                            <1>       ;mov   [edi+5], bl
9028 0000FA73 E87B000000        <1>       call   sysvideo_15_196_92
9029                            <1>              ; character pixel/forecolor,
9030                            <1>              ; 24 bpp (true colors)
9031                            <1> sysvideo_15_196_77:
9032 0000FA78 83C706            <1>       add    edi, 6
9033 0000FA7B FECC              <1>       dec    ah
9034 0000FA7D 75EB              <1>       jnz    short sysvideo_15_196_76 ; next pixel of (same) row
9035 0000FA7F 5F                <1>       pop    edi
9036 0000FA80 01EF              <1>       add    edi, ebp ; next row
9037 0000FA82 FECB              <1>       dec    bl
9038 0000FA84 75DF              <1>       jnz    short sysvideo_15_196_75 ; same pixels for next row
9039 0000FA86 46                <1>       inc    esi
9040 0000FA87 FEC9              <1>       dec    cl
9041 0000FA89 75D8              <1>       jnz    short sysvideo_15_196_74 ; next row of the character font
9042 0000FA8B C3                <1>       retn
9043                            <1>
9044                            <1> sysvideo_15_196_80:
9045                            <1> sysvideo_15_197_80:
9046                            <1>       ; 32 pixels per row
9047                            <1>       ; (12 bytes per row)
9048 0000FA8C B304              <1>       mov    bl, 4  ; 4 rows with same pixels
9049                            <1> sysvideo_15_196_81:
9050 0000FA8E 8A06              <1>       mov    al, [esi]
9051 0000FA90 57                <1>       push   edi
9052 0000FA91 B408              <1>       mov    ah, 8
9053                            <1> sysvideo_15_196_82:
9054 0000FA93 D0E8              <1>       shr    al, 1
9055 0000FA95 730A              <1>       jnc    short sysvideo_15_196_83
9056                            <1>       ; 4 pixels with same forecolor
9057                            <1>       ;mov   ebx, edx
9058                            <1>       ;shr   ebx, 16
9059                            <1>       ;mov   [edi], dx
9060                            <1>       ;mov   [edi+2], bl
9061 0000FA97 E84B000000        <1>       call   sysvideo_15_196_91
9062                            <1>       ;mov   [edi+3], dx
9063                            <1>       ;mov   [edi+5], bl
9064                            <1>       ;mov   [edi+6], dx
9065                            <1>       ;mov   [edi+8], bl
9066                            <1>       ;mov   [edi+9], dx
9067                            <1>       ;mov   [edi+11], bl
9068 0000FA9C E85A000000        <1>       call   sysvideo_15_196_93
9069                            <1>              ; character pixel/forecolor,
9070                            <1>              ; 24 bpp (true colors)
9071                            <1> sysvideo_15_196_83:
9072 0000FAA1 83C70C            <1>       add    edi, 12
9073 0000FAA4 FECC              <1>       dec    ah
9074 0000FAA6 75EB              <1>       jnz    short sysvideo_15_196_82 ; next pixel of (same) row
9075 0000FAA8 5F                <1>       pop    edi
9076 0000FAA9 01EF              <1>       add    edi, ebp ; next row
9077 0000FAAB FECB              <1>       dec    bl
9078 0000FAAD 75DF              <1>       jnz    short sysvideo_15_196_81 ; same pixels for next row
9079 0000FAAF 46                <1>       inc    esi
9080 0000FAB0 FEC9              <1>       dec    cl
9081 0000FAB2 75D8              <1>       jnz    short sysvideo_15_196_80 ; next row of the character font
9082 0000FAB4 C3                <1>       retn
9083                            <1>
9084                            <1> sysvideo_15_196_85_0:
9085                            <1> sysvideo_15_197_85_0:
9086                            <1>       ; 64 pixels per row
9087                            <1>       ; (24 bytes per row)
9088 0000FAB5 6689D3            <1>       mov    bx, dx
9089 0000FAB8 C1E210            <1>       shl    edx, 16
```

```
9090 0000FABB 6689DA              <1>       mov   dx, bx
9091                              <1> sysvideo_15_196_86:
9092 0000FABE B308                <1>       mov   bl, 8  ; 8 rows with same pixels
9093                              <1> sysvideo_15_196_87:
9094 0000FAC0 8A06                <1>       mov   al, [esi]
9095 0000FAC2 57                  <1>       push  edi
9096 0000FAC3 B408                <1>       mov   ah, 8
9097                              <1> sysvideo_15_196_88:
9098 0000FAC5 D0E8                <1>       shr   al, 1
9099 0000FAC7 730A                <1>       jnc   short sysvideo_15_196_89
9100                              <1>       ; 8 pixels with same forecolor
9101                              <1>       ;mov  ebx, edx
9102                              <1>       ;shr  ebx, 16
9103                              <1>       ;mov  [edi], dx
9104                              <1>       ;mov  [edi+2], bl
9105 0000FAC9 E819000000          <1>       call  sysvideo_15_196_91
9106                              <1>       ;mov  [edi+3], dx
9107                              <1>       ;mov  [edi+5], bl
9108                              <1>       ;mov  [edi+6], dx
9109                              <1>       ;mov  [edi+8], bl
9110                              <1>       ;mov  [edi+9], dx
9111                              <1>       ;mov  [edi+11], bl
9112                              <1>       ;mov  [edi+12], dx
9113                              <1>       ;mov  [edi+14], bl
9114                              <1>       ;mov  [edi+15], dx
9115                              <1>       ;mov  [edi+17], bl
9116                              <1>       ;mov  [edi+18], dx
9117                              <1>       ;mov  [edi+20], bl
9118                              <1>       ;mov  [edi+21], dx
9119                              <1>       ;mov  [edi+23], bl
9120 0000FACE E838000000          <1>       call  sysvideo_15_196_94
9121                              <1>             ; character pixel/forecolor,
9122                              <1>             ; 24 bpp (true colors)
9123                              <1> sysvideo_15_196_89:
9124 0000FAD3 83C718              <1>       add   edi, 24
9125 0000FAD6 FECC                <1>       dec   ah
9126 0000FAD8 75EB                <1>       jnz   short sysvideo_15_196_88 ; next pixel of (same) row
9127 0000FADA 5F                  <1>       pop   edi
9128 0000FADB 01EF                <1>       add   edi, ebp ; next row
9129 0000FADD FECB                <1>       dec   bl
9130 0000FADF 75DF                <1>       jnz   short sysvideo_15_196_87 ; same pixels for next row
9131 0000FAE1 46                  <1>       inc   esi
9132 0000FAE2 FEC9                <1>       dec   cl
9133 0000FAE4 75D8                <1>       jnz   short sysvideo_15_196_86 ; next row of the character font
9134 0000FAE6 C3                  <1>       retn
9135                              <1>
9136                              <1> sysvideo_15_196_91:
9137                              <1>       ; 02/01/2021
9138 0000FAE7 89D3                <1>       mov   ebx, edx
9139 0000FAE9 C1EB10              <1>       shr   ebx, 16
9140 0000FAEC 668917              <1>       mov   [edi], dx
9141 0000FAEF 885F02              <1>       mov   [edi+2], bl
9142 0000FAF2 C3                  <1>       retn
9143                              <1> sysvideo_15_196_92:
9144 0000FAF3 66895703            <1>       mov   [edi+3], dx
9145 0000FAF7 885F05              <1>       mov   [edi+5], bl
9146 0000FAFA C3                  <1>       retn
9147                              <1> sysvideo_15_196_93:
9148 0000FAFB 66895706            <1>       mov   [edi+6], dx
9149 0000FAFF 885F08              <1>       mov   [edi+8], bl
9150 0000FB02 66895709            <1>       mov   [edi+9], dx
9151 0000FB06 885F0B              <1>       mov   [edi+11], bl
9152 0000FB09 EBE8                <1>       jmp   short sysvideo_15_196_92
9153                              <1> sysvideo_15_196_94:
9154 0000FB0B 6689570C            <1>       mov   [edi+12], dx
9155 0000FB0F 885F0E              <1>       mov   [edi+14], bl
9156 0000FB12 6689570F            <1>       mov   [edi+15], dx
9157 0000FB16 885F11              <1>       mov   [edi+17], bl
9158 0000FB19 66895712            <1>       mov   [edi+18], dx
9159 0000FB1D 885F14              <1>       mov   [edi+20], bl
9160 0000FB20 66895715            <1>       mov   [edi+21], dx
9161 0000FB24 885F17              <1>       mov   [edi+23], bl
9162 0000FB27 EBD2                <1>       jmp   short sysvideo_15_196_93
9163                              <1>
9164                              <1> sysvideo_15_200:
9165                              <1>       ; 29/12/2020
9166 0000FB29 6639EA              <1>       cmp   dx, bp ; end column < screen width ?
9167 0000FB2C 7205                <1>       jb    short sysvideo_15_201
9168                              <1>       ; fix bottom right position of the window
9169 0000FB2E 6689EA              <1>       mov   dx, bp
9170 0000FB31 664A                <1>       dec   dx
9171                              <1> sysvideo_15_201:
9172 0000FB33 C3                  <1>       retn
9173                              <1>
9174                              <1> sysvideo_15_202:
9175                              <1>       ; 30/12/2020
9176 0000FB34 89EB                <1>       mov   ebx, ebp ; screen width
9177                              <1> sysvideo_15_203:
9178 0000FB36 89C8                <1>       mov   eax, ecx
9179 0000FB38 C1E810              <1>       shr   eax, 16  ; top row
9180 0000FB3B F7E3                <1>       mul   ebx
9181 0000FB3D 6689CA              <1>       mov   dx, cx ; top left column
9182 0000FB40 01D0                <1>       add   eax, edx
9183 0000FB42 C3                  <1>       retn
9184                              <1>
9185                              <1> sysvideo_15_204:
9186                              <1>       ; 30/12/2020
9187 0000FB43 010D[64030300]      <1>       add   [u.r0], ecx
9188 0000FB49 01DF                <1>       add   edi, ebx ; next row
9189 0000FB4B 01CE                <1>       add   esi, ecx
9190 0000FB4D 3B3C24              <1>       cmp   edi, [esp] ; stop addr (included in loop)
9191 0000FB50 C3                  <1>       retn
9192                              <1>
9193                              <1> sysvideo_39:
9194                              <1>       ; 04/01/2021
```

```
9195                                    <1>       ; 03/01/2021
9196                                    <1>       ; 23/11/2020
9197                                    <1>       ; BH = 3
9198                                    <1>       ; PIXEL READ/WRITE
9199                                    <1>
9200                                    <1>       ; 04/01/2021 (TRDOS 386 v2.0.3)
9201 0000FB51 80FB03                   <1>       cmp   bl, 3
9202 0000FB54 7610                     <1>       jna   short sysvideo_39_1
9203                                    <1> sysvideo_39_0:
9204                                    <1>       ; error
9205 0000FB56 B3FF                     <1>       mov   bl, 0FFh
9206 0000FB58 8B2D[60030300]           <1>       mov   ebp, [u.usp]  ; ebp points to user's registers
9207 0000FB5E 895D10                   <1>       mov   [ebp+16], ebx ; EBX
9208 0000FB61 E986DDFFFF               <1>       jmp   sysret
9209                                    <1> sysvideo_39_1:
9210 0000FB66 803D[BA6F0000]FF         <1>       cmp   byte [CRT_MODE], 0FFh
9211 0000FB6D 7312                     <1>       jnb   short sysvideo_39_2 ; SVGA (VESA VBE) video mode
9212                                    <1>
9213                                    <1>       ; Std VGA or CGA mode
9214 0000FB6F 81C200000A00             <1>       add   edx, 0A0000h
9215 0000FB75 72DF                     <1>       jc    short sysvideo_39_0
9216 0000FB77 81FAFFFF0A00             <1>       cmp   edx, 0AFFFFh
9217 0000FB7D 77D7                     <1>       ja    short sysvideo_39_0
9218 0000FB7F EB1E                     <1>       jmp   short sysvideo_39_3 ; 8bpp
9219                                    <1>
9220                                    <1> sysvideo_39_2:
9221                                    <1>       ; use current vbe (svga) video mode
9222                                    <1>
9223                                    <1>       ; get LFB address
9224 0000FB81 A1[2C120300]             <1>       mov   eax, [LFB_ADDR] ; [LFB_Info+LFBINFO.LFB_addr]
9225 0000FB86 09C0                     <1>       or    eax, eax
9226 0000FB88 74CC                     <1>       jz    short sysvideo_39_0
9227 0000FB8A 3B15[30120300]           <1>       cmp   edx, [LFB_SIZE]
9228 0000FB90 73C4                     <1>       jnb   short sysvideo_39_0
9229                                    <1>
9230 0000FB92 01C2                     <1>       add   edx, eax
9231                                    <1>       ;jc   short sysvideo_39_0
9232                                    <1>
9233                                    <1>       ; Pixel read/write in VESA VBE (2/3) video mode
9234                                    <1>       ; Video memory at Linear Frame Buffer base address
9235                                    <1>
9236 0000FB94 8A3D[38120300]           <1>       mov   bh, [LFB_Info+LFBINFO.bpp]
9237                                    <1>
9238 0000FB9A 80FF08                   <1>       cmp   bh, 8 ; 8bpp
9239 0000FB9D 775D                     <1>       ja    short sysvideo_39_17
9240                                    <1>
9241                                    <1>       ; 8 bits per pixel
9242                                    <1> sysvideo_39_3:
9243 0000FB9F 80FB01                   <1>       cmp   bl, 1 ; 1 = write pixel
9244 0000FBA2 7406                     <1>       je    short sysvideo_39_5
9245 0000FBA4 7712                     <1>       ja    short sysvideo_39_8
9246                                    <1> sysvideo_39_4:
9247                                    <1>       ; read pixel (8bpp)
9248 0000FBA6 8A02                     <1>       mov   al, [edx]
9249                                    <1>       ;mov  [u.r0], al
9250                                    <1>       ;jmp  sysret
9251 0000FBA8 EB04                     <1>       jmp   short sysvideo_39_7
9252                                    <1> sysvideo_39_5:
9253                                    <1>       ; write pixel (8bpp)
9254 0000FBAA 88C8                     <1>       mov   al, cl
9255                                    <1> sysvideo_39_6:
9256 0000FBAC 8802                     <1>       mov   [edx], al
9257                                    <1> sysvideo_39_7:
9258 0000FBAE A2[64030300]             <1>       mov   [u.r0], al
9259 0000FBB3 E934DDFFFF               <1>       jmp   sysret
9260                                    <1> sysvideo_39_8:
9261 0000FBB8 80FB03                   <1>       cmp   bl, 3 ; mix
9262 0000FBBB 7208                     <1>       jb    short sysvideo_39_9
9263                                    <1>       ; mix  pixel colors (8bpp)
9264 0000FBBD 8A02                     <1>       mov   al, [edx]
9265 0000FBBF 00C8                     <1>       add   al, cl
9266 0000FBC1 D0D8                     <1>       rcr   al, 1
9267 0000FBC3 EBE7                     <1>       jmp   short sysvideo_39_6
9268                                    <1> sysvideo_39_9:
9269                                    <1>        ; swap pixel colors (8bpp)
9270 0000FBC5 88C8                     <1>       mov   al, cl
9271 0000FBC7 8602                     <1>       xchg  [edx], al
9272 0000FBC9 EBE3                     <1>       jmp   short sysvideo_39_7
9273                                    <1>
9274                                    <1>       ; 16 bits per pixel
9275                                    <1> sysvideo_39_10:
9276 0000FBCB 80FB01                   <1>       cmp   bl, 1 ; 1 = write pixel
9277 0000FBCE 7406                     <1>       je    short sysvideo_39_12
9278 0000FBD0 7714                     <1>       ja    short sysvideo_39_15
9279                                    <1> sysvideo_39_11:
9280                                    <1>       ; read pixel (16bpp)
9281 0000FBD2 8B02                     <1>       mov   eax, [edx]
9282                                    <1>       ;mov  [u.r0], ax
9283                                    <1>       ;jmp  sysret
9284 0000FBD4 EB05                     <1>       jmp   short sysvideo_39_14
9285                                    <1> sysvideo_39_12:
9286                                    <1>       ; write pixel (16bpp)
9287 0000FBD6 89C8                     <1>       mov   eax, ecx
9288                                    <1> sysvideo_39_13:
9289 0000FBD8 668902                   <1>       mov   [edx], ax
9290                                    <1> sysvideo_39_14:
9291 0000FBDB 66A3[64030300]           <1>       mov   [u.r0], ax
9292 0000FBE1 E906DDFFFF               <1>       jmp   sysret
9293                                    <1> sysvideo_39_15:
9294 0000FBE6 80FB03                   <1>       cmp   bl, 3 ; mix
9295 0000FBE9 720A                     <1>       jb    short sysvideo_39_16
9296                                    <1>       ; mix  pixel colors (16bpp)
9297 0000FBEB 8B02                     <1>       mov   eax, [edx]
9298 0000FBED 6601C8                   <1>       add   ax, cx
9299 0000FBF0 66D1D8                   <1>       rcr   ax, 1
```

```
9300 0000FBF3 EBE3              <1>       jmp    short sysvideo_39_13
9301                            <1> sysvideo_39_16:
9302                            <1>        ; swap pixel colors (16bpp)
9303 0000FBF5 89C8              <1>       mov    eax, ecx
9304 0000FBF7 668702            <1>       xchg   [edx], ax
9305 0000FBFA EBDF              <1>       jmp    short sysvideo_39_14
9306                            <1> sysvideo_39_17:
9307 0000FBFC 80FF18            <1>       cmp    bh, 24
9308 0000FBFF 7743              <1>       ja     short sysvideo_39_24
9309 0000FC01 72C8              <1>       jb     short sysvideo_39_10
9310                            <1>
9311                            <1>       ; 24 bits per pixel
9312 0000FC03 81E1FFFFFF00      <1>       and    ecx, 0FFFFFFh
9313 0000FC09 80FB01            <1>       cmp    bl, 1 ; 1 = write pixel
9314 0000FC0C 7406              <1>       je     short sysvideo_39_19
9315 0000FC0E 7712              <1>       ja     short sysvideo_39_22
9316                            <1> sysvideo_39_18:
9317                            <1>       ; read pixel (24bpp)
9318 0000FC10 8B02              <1>       mov    eax, [edx]
9319                            <1>       ;and   eax, 0FFFFFFh
9320                            <1>       ;mov   [u.r0], eax
9321                            <1>       ;jmp   sysret
9322 0000FC12 EB04              <1>       jmp    short sysvideo_39_21
9323                            <1> sysvideo_39_19:
9324                            <1>       ; write pixel (24bpp)
9325 0000FC14 89C8              <1>       mov    eax, ecx
9326                            <1> sysvideo_39_20:
9327                            <1>       ;and   eax, 0FFFFFFh
9328 0000FC16 8902              <1>       mov    [edx], eax
9329                            <1> sysvideo_39_21:
9330 0000FC18 A3[64030300]      <1>       mov    [u.r0], eax
9331 0000FC1D E9CADCFFFF        <1>       jmp    sysret
9332                            <1> sysvideo_39_22:
9333 0000FC22 80FB03            <1>       cmp    bl, 3 ; mix
9334 0000FC25 720D              <1>       jb     short sysvideo_39_23
9335                            <1>       ; mix  pixel colors (24bpp)
9336 0000FC27 8B02              <1>       mov    eax, [edx]
9337 0000FC29 25FFFFFF00        <1>       and    eax, 0FFFFFFh
9338                            <1>       ;and   ecx, 0FFFFFFh
9339 0000FC2E 01C8              <1>       add    eax, ecx
9340 0000FC30 D1D8              <1>       rcr    eax, 1
9341 0000FC32 EBE2              <1>       jmp    short sysvideo_39_20
9342                            <1> sysvideo_39_23:
9343                            <1>        ; swap pixel colors (24bpp)
9344 0000FC34 89C8              <1>       mov    eax, ecx
9345                            <1>       ;and   eax, 0FFFFFFh
9346 0000FC36 668702            <1>       xchg   [edx], ax
9347 0000FC39 C1C810            <1>       ror    eax, 16
9348 0000FC3C 884202            <1>       mov    [edx+2], al
9349 0000FC3F C1C010            <1>       rol    eax, 16
9350 0000FC42 EBD4              <1>       jmp    short sysvideo_39_21
9351                            <1>
9352                            <1>       ; 32 bits per pixel
9353                            <1> sysvideo_39_24:
9354 0000FC44 80FB01            <1>       cmp    bl, 1 ; 1 = write pixel
9355 0000FC47 7406              <1>       je     short sysvideo_39_26
9356 0000FC49 7712              <1>       ja     short sysvideo_39_29
9357                            <1> sysvideo_39_25:
9358                            <1>       ; read pixel (32bpp)
9359 0000FC4B 8B02              <1>       mov    eax, [edx]
9360                            <1>       ;mov   [u.r0], eax
9361                            <1>       ;jmp   sysret
9362 0000FC4D EB04              <1>       jmp    short sysvideo_39_28
9363                            <1> sysvideo_39_26:
9364                            <1>       ; write pixel (32bpp)
9365 0000FC4F 89C8              <1>       mov    eax, ecx
9366                            <1> sysvideo_39_27:
9367 0000FC51 8902              <1>       mov    [edx], eax
9368                            <1> sysvideo_39_28:
9369 0000FC53 A3[64030300]      <1>       mov    [u.r0], eax
9370 0000FC58 E98FDCFFFF        <1>       jmp    sysret
9371                            <1> sysvideo_39_29:
9372 0000FC5D 80FB03            <1>       cmp    bl, 3 ; mix
9373 0000FC60 7208              <1>       jb     short sysvideo_39_30
9374                            <1>       ; mix  pixel colors (32bpp)
9375 0000FC62 8B02              <1>       mov    eax, [edx]
9376 0000FC64 01C8              <1>       add    eax, ecx
9377 0000FC66 D1D8              <1>       rcr    eax, 1
9378 0000FC68 EBE7              <1>       jmp    short sysvideo_39_27
9379                            <1> sysvideo_39_30:
9380                            <1>        ; swap pixel colors (32bpp)
9381 0000FC6A 89C8              <1>       mov    eax, ecx
9382 0000FC6C 8702              <1>       xchg   [edx], eax
9383 0000FC6E EBE3              <1>       jmp    short sysvideo_39_28
9384                            <1>
9385                            <1> sysvideo_16:
9386                            <1>       ; 23/11/2020
9387 0000FC70 80FF04            <1>       cmp    bh, 4
9388 0000FC73 0F82D8FEFFFF      <1>       jb     sysvideo_39 ; bh = 3, pixel r/w
9389 0000FC79 7721              <1>       ja     short sysvideo_17
9390                            <1>
9391                            <1>       ; BH = 4
9392                            <1>       ; Direct User Access for CGA video memory.
9393                            <1>       ; Setup user's page tables for direct access to 0B8000h.
9394                            <1>       ;
9395                            <1>       ; Permission checks are not implemented yet !
9396                            <1>       ; (11/07/2016)
9397                            <1>
9398 0000FC7B B800800B00        <1>       mov    eax, 0B8000h
9399 0000FC80 B908000000        <1>       mov    ecx, 8 ; 8 pages (8*4K=32K)
9400 0000FC85 89C3              <1>       mov    ebx, eax ; 12/05/2017 ; virtual = physical
9401 0000FC87 E8646AFFFF        <1>       call   direct_memory_access
9402 0000FC8C 0F825ADCFFFF      <1>       jc     sysret
9403                            <1>       ; eax = 0B8000h if there is not an error
9404 0000FC92 A3[64030300]      <1>       mov    [u.r0], eax
```

```
9405 0000FC97 E950DCFFFF          <1>       jmp    sysret
9406                              <1>
9407                              <1> sysvideo_17:
9408                              <1>       ; 23/12/2020
9409                              <1>       ; 11/12/2020
9410                              <1>       ; 10/12/2020
9411                              <1>       ; 23/11/2020
9412 0000FC9C 80FF06              <1>       cmp    bh, 6
9413 0000FC9F 740B                <1>       je     short sysvideo_17_0
9414 0000FCA1 0F82B9000000        <1>       jb     sysvideo_18
9415 0000FCA7 E913010000          <1>       jmp    sysvideo_20 ; ja
9416                              <1>
9417                              <1> sysvideo_17_0:
9418                              <1>       ; BH = 6
9419                              <1>       ; Direct User Access to Linear Frame Buffer.
9420                              <1>       ; Setup user's page tables for direct access to LFB.
9421                              <1>       ;
9422                              <1>       ; Permission checks are not implemented yet !
9423                              <1>       ; (10/12/2020)
9424                              <1>
9425 0000FCAC 80FBFF              <1>       cmp    bl, 0FFh ; current video mode
9426 0000FCAF 722C                <1>       jb     short sysvideo_17_2 ; for desired video mode
9427                              <1>
9428 0000FCB1 381D[BA6F0000]      <1>       cmp    [CRT_MODE], bl ; VESA VBE video mode ?
9429 0000FCB7 750E                <1>       jne    short sysvideo_17_1
9430 0000FCB9 668B0D[1E120300]    <1>       mov    cx, [video_mode]
9431 0000FCC0 6681E1FF01          <1>       and    cx, 1FFh
9432 0000FCC5 EB29                <1>       jmp    short sysvideo_17_3
9433                              <1> sysvideo_17_1:
9434                              <1>       ; 11/12/2020
9435 0000FCC7 88DF                <1>       mov    bh, bl ; 0FFh
9436 0000FCC9 8A1D[BA6F0000]      <1>       mov    bl, [CRT_MODE] ; VGA/CGA video mode
9437 0000FCCF 8B2D[60030300]      <1>       mov    ebp, [u.usp]  ; ebp points to user's registers
9438                              <1>       ; 23/12/2020
9439 0000FCD5 895D10              <1>       mov    [ebp+16], ebx ; return to user with EBX value
9440 0000FCD8 E90FDCFFFF          <1>       jmp    sysret ; return to user with EAX = 0
9441                              <1> sysvideo_17_2:
9442                              <1>       ; bl = VESA video mode - 100h
9443 0000FCDD B701                <1>       mov    bh, 1 ; bx = 1XXh
9444 0000FCDF 53                  <1>       push   ebx ; requested vesa video mode
9445 0000FCE0 E8BC3EFFFF          <1>       call   vbe_biosfn_return_current_mode
9446 0000FCE5 59                  <1>       pop    ecx ; requested vesa video mode
9447 0000FCE6 6681E3FF01          <1>       and    bx, 1FFh
9448 0000FCEB 6639D9              <1>       cmp    cx, bx
9449 0000FCEE 7564                <1>       jne    short sysvideo_17_8
9450                              <1> sysvideo_17_3:
9451 0000FCF0 663B0D[2A120300]    <1>       cmp    cx, [LFB_Info+LFBINFO.mode]
9452 0000FCF7 755B                <1>       jne    short sysvideo_17_8
9453                              <1> sysvideo_17_4:
9454                              <1>       ; 11/12/2020
9455 0000FCF9 A1[2C120300]        <1>       mov    eax, [LFB_Info+LFBINFO.LFB_addr]
9456                              <1>       ; 21/12/2020
9457 0000FCFE 09C0                <1>       or     eax, eax
9458 0000FD00 744D                <1>       jz     short sysvideo_17_7
9459                              <1>       ;
9460 0000FD02 8B0D[30120300]      <1>       mov    ecx, [LFB_Info+LFBINFO.LFB_size] ; buff size
9461 0000FD08 89C3                <1>       mov    ebx, eax ; user's address = physical address
9462                              <1>       ;push  ebx
9463 0000FD0A 51                  <1>       push   ecx
9464                              <1>       ; 21/12/2020
9465 0000FD0B 81C1FF0F0000        <1>       add    ecx, 4095  ; PAGESIZE - 1
9466                              <1>       ; 14/12/2020
9467 0000FD11 C1E90C              <1>       shr    ecx, 12  ; convert bytes to pages
9468 0000FD14 E8D769FFFF          <1>       call   direct_memory_access
9469 0000FD19 5A                  <1>       pop    edx  ; linear frame buffer size in bytes
9470                              <1>       ;pop   eax  ; linear frame buffer address (physical)
9471 0000FD1A 7233                <1>       jc     short sysvideo_17_7 ; [u.r0] = eax = 0
9472                              <1> sysvideo_17_5:
9473 0000FD1C 668B0D[36120300]    <1>       mov    cx, [LFB_Info+LFBINFO.Y_res] ; screen height
9474 0000FD23 C1E110              <1>       shl    ecx, 16
9475 0000FD26 668B0D[34120300]    <1>       mov    cx,  [LFB_Info+LFBINFO.X_res] ; screen width
9476 0000FD2D 31DB                <1>       xor    ebx, ebx
9477 0000FD2F 8A1D[38120300]      <1>       mov    bl, [LFB_Info+LFBINFO.bpp] ; bits per pixel
9478 0000FD35 8A3D[2A120300]      <1>       mov    bh, [LFB_Info+LFBINFO.mode] ; XX part of 1XXh
9479                              <1> sysvideo_26_4: ; 23/12/2020
9480 0000FD3B 8B2D[60030300]      <1>       mov    ebp, [u.usp]  ; ebp points to user's registers
9481 0000FD41 895514              <1>       mov    [ebp+20], edx ; return to user with EDX value
9482 0000FD44 895D10              <1>       mov    [ebp+16], ebx ; EBX
9483 0000FD47 894D18              <1>       mov    [ebp+24], ecx ; ECX
9484                              <1> sysvideo_17_6:
9485 0000FD4A A3[64030300]        <1>       mov    [u.r0], eax ; LFB address
9486                              <1> sysvideo_17_7:
9487 0000FD4F E998DBFFFF          <1>       jmp    sysret
9488                              <1> sysvideo_17_8:
9489                              <1>       ; cx = mode
9490                              <1>       ; 21/12/2020
9491 0000FD54 80CD40              <1>       or     ch, 40h  ; Linear frame buffer flag
9492 0000FD57 E8673CFFFF          <1>       call   _vbe_biosfn_return_mode_info
9493 0000FD5C 72F1                <1>       jc     short sysvideo_17_7
9494 0000FD5E EB99                <1>       jmp    short sysvideo_17_4
9495                              <1>
9496                              <1> sysvideo_18:
9497                              <1>       ; BH = 5
9498                              <1>       ; Direct User Access for VGA video memory.
9499                              <1>       ; Setup user's page tables for direct access to 0A0000h.
9500                              <1>       ;
9501                              <1>       ; Permission checks are not implemented yet !
9502                              <1>       ; (11/07/2016)
9503                              <1>
9504 0000FD60 B800000A00          <1>       mov    eax, 0A0000h
9505 0000FD65 B910000000          <1>       mov    ecx, 16 ; 16 pages (16*4K=64K)
9506 0000FD6A 89C3                <1>       mov    ebx, eax ; 12/05/2017 ; virtual = physical
9507 0000FD6C E87F69FFFF          <1>       call   direct_memory_access
9508 0000FD71 0F8275DBFFFF        <1>       jc     sysret
9509                              <1>       ; eax = 0A0000h if there is not an error
```

```
9510 0000FD77 A3[64030300]        <1>       mov    [u.r0], eax
9511 0000FD7C E96BDBFFFF          <1>       jmp    sysret
9512                              <1>
9513                              <1> sysvideo_19:
9514                              <1>       ; 22/01/2021
9515                              <1>       ; 12/12/2020
9516                              <1>       ; 11/12/2020
9517                              <1>       ; 23/11/2020
9518                              <1>       ; BH = 7
9519                              <1>       ; Get (Super/Extended VGA) mode
9520                              <1>       ; and Linear Frame Buffer info.
9521                              <1>
9522                              <1>       ; 22/01/2021
9523 0000FD81 B3FF                <1>       mov    bl, 0FFh
9524                              <1>       ; 11/12/2020
9525                              <1>       ;cmp   byte [CRT_MODE], 0FFh ; (extended mode?)
9526                              <1>       ; 22/01/2021
9527 0000FD83 381D[BA6F0000]      <1>       cmp    [CRT_MODE], bl  ; 0FFh
9528                              <1>       ;jb    sysvideo_17_1; not a VESA VBE mode
9529                              <1>       ; 12/12/2020
9530 0000FD89 7305                <1>       jnb    short sysvideo_19_0
9531 0000FD8B E937FFFFFF          <1>       jmp    sysvideo_17_1
9532                              <1>
9533                              <1> sysvideo_19_0:
9534 0000FD90 E80C3EFFFF          <1>       call   vbe_biosfn_return_current_mode
9535 0000FD95 6681E3FF01          <1>       and    bx, 1FFh
9536 0000FD9A 663B1D[2A120300]    <1>       cmp    bx, [LFB_Info+LFBINFO.mode]
9537 0000FDA1 7510                <1>       jne    short sysvideo_19_2
9538                              <1> sysvideo_19_1:
9539 0000FDA3 A1[2C120300]        <1>       mov    eax, [LFB_Info+LFBINFO.LFB_addr]
9540 0000FDA8 8B15[30120300]      <1>       mov    edx, [LFB_Info+LFBINFO.LFB_size]
9541 0000FDAE E969FFFFFF          <1>       jmp    sysvideo_17_5
9542                              <1> sysvideo_19_2:
9543 0000FDB3 E80B3CFFFF          <1>       call   _vbe_biosfn_return_mode_info
9544 0000FDB8 73E9                <1>       jnc    short sysvideo_19_1
9545 0000FDBA E92DDBFFFF          <1>       jmp    sysret
9546                              <1>
9547                              <1> sysvideo_20:
9548                              <1>       ; 11/12/2020
9549                              <1>       ; 23/11/2020
9550 0000FDBF 80FF08              <1>       cmp    bh, 8
9551 0000FDC2 72BD                <1>       jb     short sysvideo_19 ; video mode & lfb info
9552 0000FDC4 0F8780000000        <1>       ja     sysvideo_21 ; 12/12/2020
9553                              <1>
9554                              <1>       ; BH = 8
9555                              <1>       ; Set (Super/Extended VGA) mode & return LFB info
9556                              <1>       ;
9557                              <1>
9558                              <1>       ; 11/12/2020
9559 0000FDCA 80FBFF              <1>       cmp    bl, 0FFh  ; CGA/VGA mode ?
9560 0000FDCD 7318                <1>       jnb    short sysvideo_20_1
9561                              <1>
9562                              <1>       ;xor   ah, ah
9563 0000FDCF 88D8                <1>       mov    al, bl
9564                              <1> sysvideo_20_0:
9565 0000FDD1 E89319FFFF          <1>       call   _int10h  ; uses vbe3 pmi32 option
9566 0000FDD6 83F8FF              <1>       cmp    eax, 0FFFFFFFFh ; -1
9567 0000FDD9 7459                <1>       je     short sysvideo_20_3 ; error
9568                              <1>
9569                              <1>       ; 11/12/2020
9570                              <1>       ; alternative (it does not use vbe3 pmi32)
9571                              <1>       ;push  eax
9572                              <1>       ;call  _set_mode
9573                              <1>       ;pop   eax
9574                              <1>       ;jc    short sysvideo_20_3
9575                              <1>
9576                              <1>       ;inc   eax
9577 0000FDDB FEC0                <1>       inc    al
9578                              <1>       ;mov   [u.r0], ax ; video mode + 1
9579 0000FDDD A2[64030300]        <1>       mov    [u.r0], al
9580 0000FDE2 E905DBFFFF          <1>       jmp    sysret
9581                              <1>
9582                              <1> sysvideo_20_1:
9583                              <1>       ; cx = vesa video mode
9584 0000FDE7 6689C8              <1>       mov    ax, cx
9585 0000FDEA 663D0001            <1>       cmp    ax, 100h
9586 0000FDEE 72E1                <1>       jb     short sysvideo_20_0 ; VGA/CGA mode
9587 0000FDF0 663DFF01            <1>       cmp    ax, 1FFh
9588                              <1>       ;ja    short sysvideo_20_4 ; not valid
9589 0000FDF4 773E                <1>       ja     short sysvideo_20_3
9590 0000FDF6 50                  <1>       push   eax
9591 0000FDF7 6689C3              <1>       mov    bx, ax
9592 0000FDFA 66B8024F            <1>       mov    ax, 4F02h
9593                              <1>
9594                              <1>       ; simulate _int10h (int 31h) for func 4F02h
9595                              <1>       ;pushfd
9596                              <1>       ;push  cs
9597                              <1>       ;push  sysvideo_20_1_retn
9598                              <1>       ;push  es ; *
9599                              <1>       ;push  ds ; **      ; SAVE WORK AND PARAMETER REGISTERS
9600                              <1>       ;jmp   VBE_func
9601                              <1> ;sysvideo_20_1_retn:
9602                              <1>
9603 0000FDFE E86619FFFF          <1>       call   _int10h ; simulate int 10h (int 31h)
9604                              <1>
9605 0000FE03 6683F84F            <1>       cmp    ax, 004Fh
9606 0000FE07 58                  <1>       pop    eax
9607 0000FE08 752A                <1>       jne    short sysvideo_20_3 ; error
9608                              <1>       ;pop   eax
9609 0000FE0A 40                  <1>       inc    eax
9610 0000FE0B A3[64030300]        <1>       mov    [u.r0], eax ; video mode + 1
9611 0000FE10 09D2                <1>       or     edx, edx ; is LFBINFO requested by user ?
9612                              <1>       ;jz    short sysvideo_20_4
9613 0000FE12 7420                <1>       jz     short sysvideo_20_3 ; no
9614                              <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>
0000FE14 52                     <1>       push   edx
0000FE15 48                     <1>       dec    eax  ; video mode
0000FE16 BE[2A120300]           <1>       mov    esi, LFB_Info
0000FE1B 663B06                 <1>       cmp    ax, [esi+LFBINFO.mode]
0000FE1E 7407                   <1>       je     short sysvideo_20_2
                                <1>
0000FE20 E89E3BFFFF             <1>       call   _vbe_biosfn_return_mode_info
                                <1>       ;jnc   short sysvideo_20_2
0000FE25 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
0000FE27 5F                     <1>       pop    edi
0000FE28 B910000000             <1>       mov    ecx, LFBINFO.size ; 16
0000FE2D E892140000             <1>       call   transfer_to_user_buffer ; fast transfer
0000FE32 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:
0000FE34 E9B3DAFFFF             <1>       jmp    sysret
                                <1>
                                <1> sysvideo_20_4:
0000FE39 5A                     <1>       pop    edx
                                <1> sysvideo_20_5:
0000FE3A 31D2                   <1>       xor    edx, edx ; 0
                                <1>       ; edx = 0 -> invalid LFBINFO data
0000FE3C 8B2D[60030300]         <1>       mov    ebp, [u.usp]  ; ebp points to user's registers
0000FE42 895514                 <1>       mov    [ebp+20], edx ; return to user with EDX value
0000FE45 E9A2DAFFFF             <1>       jmp    sysret
                                <1>
                                <1> sysvideo_21:
                                <1>       ; 04/01/2021
                                <1>       ; 03/12/2020
0000FE4A 80FF0A                 <1>       cmp    bh, 10
0000FE4D 0F82A8010000           <1>       jb     sysvideo_22 ; VESA VBE3 pmi parms
                                <1>       ; 23/12/2020
0000FE53 0F8432020000           <1>       je     sysvideo_26 ; Video memory mapping
                                <1>
                                <1>       ; 04/01/2020
0000FE59 80FF0B                 <1>       cmp    bh, 11
0000FE5C 0F87B4020000           <1>       ja     sysvideo_27
                                <1>
                                <1>       ; BH = 11
                                <1>       ; set/read DAC color registers (for 8bpp)
                                <1>
0000FE62 80FB04                 <1>       cmp    bl, 4
0000FE65 0F83AB000000           <1>       jnb    sysvideo_21_7; BMP file type palette
                                <1>                            ; handling
0000FE6B F6C301                 <1>       test   bl, 1
0000FE6E 7555                   <1>       jnz    short sysvideo_21_4 ; set/write DAC colors
                                <1>
                                <1>       ; Read DAC color register or all DAC color registers
0000FE70 F6C302                 <1>       test   bl, 2  ; read single DAC color register
0000FE73 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>
0000FE75 66BAC703               <1>       mov    dx, 3C7h ; VGAREG_DAC_READ_ADDRESS
0000FE79 88C8                   <1>       mov    al, cl ; DAC color register
0000FE7B 31C9                   <1>       xor    ecx, ecx ; (this may not be necessary)
0000FE7D EE                     <1>       out    dx, al
                                <1>       ;mov   dx, 3C9h ; VGAREG_DAC_DATA
0000FE7E B2C9                   <1>       mov    dl, 0C9h
0000FE80 EC                     <1>       in     al, dx
0000FE81 88C4                   <1>       mov    ah, al ; red
0000FE83 EC                     <1>       in     al, dx
0000FE84 88C1                   <1>       mov    cl, al ; green
0000FE86 EC                     <1>       in     al, dx
0000FE87 88C5                   <1>       mov    ch, al ; blue
0000FE89 C1E108                 <1>       shl    ecx, 8
0000FE8C 88E1                   <1>       mov    cl, ah ; red
                                <1>       ; CL = Red, CH = Green, byte 3 = Blue, byte 4 = 0
                                <1> sysvideo_21_0:
0000FE8E 890D[64030300]         <1>       mov    [u.r0], ecx
                                <1> sysvideo_21_1:
0000FE94 E953DAFFFF             <1>       jmp    sysret
                                <1> sysvideo_21_2:
                                <1>       ; read all DAC color registers
0000FE99 89CB                   <1>       mov    ebx, ecx ; user's buffer address
0000FE9B BF00600900             <1>       mov    edi, VBE3STACKADDR
0000FEA0 89FE                   <1>       mov    esi, edi
0000FEA2 B900030000             <1>       mov    ecx, 768 ; 256*3
0000FEA7 51                     <1>       push   ecx
0000FEA8 66BAC703               <1>       mov    dx, 3C7h ; VGAREG_DAC_READ_ADDRESS
0000FEAC 28C0                   <1>       sub    al, al ; 0
0000FEAE EE                     <1>       out    dx, al
                                <1>       ;mov   dx, 3C9h ; VGAREG_DAC_DATA
0000FEAF B2C9                   <1>       mov    dl, 0C9h
                                <1> sysvideo_21_3:
```

```
9720 0000FEB1 EC                <1>       in     al, dx
9721 0000FEB2 AA                <1>       stosb
9722 0000FEB3 EC                <1>       in     al, dx
9723 0000FEB4 AA                <1>       stosb
9724 0000FEB5 EC                <1>       in     al, dx
9725 0000FEB6 AA                <1>       stosb
9726 0000FEB7 E2F8              <1>       loop   sysvideo_21_3
9727 0000FEB9 59                <1>       pop    ecx
9728                            <1>
9729 0000FEBA 89DF              <1>       mov    edi, ebx ; user's buffer address
9730                            <1>       ;mov   esi, VBE3STACKADDR
9731                            <1>       ;mov   ecx, 256*3 = 768
9732 0000FEBC E803140000        <1>       call   transfer_to_user_buffer
9733 0000FEC1 72D1              <1>       jc     short sysvideo_21_1
9734                            <1>       ;mov   [u.r0], ecx ; actual transfer count
9735 0000FEC3 EBC9              <1>       jmp    short sysvideo_21_0
9736                            <1>
9737                            <1> sysvideo_21_4:
9738                            <1>       ; Set/Write DAC color register or all registers
9739 0000FEC5 F6C302            <1>       test   bl, 2 ; write/set single DAC color register
9740 0000FEC8 741C              <1>       jz     short sysvideo_21_5 ; set all DAC color regs
9741                            <1>
9742                            <1>       ; set single DAC color register
9743                            <1>       ; CL = DAC color register (index)
9744                            <1>       ; (byte 1 = Red, byte 2 = Green, byte 3 = Blue)
9745                            <1>
9746 0000FECA 66BAC803          <1>       mov    dx, 3C8h ; VGAREG_DAC_WRITE_ADDRESS
9747 0000FECE 89C8              <1>       mov    eax, ecx ; DAC color register (index)
9748 0000FED0 C1E910            <1>       shr    ecx, 16  ; cl = green, AH = Red
9749 0000FED3 EE                <1>       out    dx, al
9750                            <1>       ;mov   dx, 3C9h ; VGAREG_DAC_DATA
9751 0000FED4 FEC2              <1>       inc    dl
9752 0000FED6 88E0              <1>       mov    al, ah ; Red
9753 0000FED8 EE                <1>       out    dx, al
9754 0000FED9 88C8              <1>       mov    al, cl ; Green
9755 0000FEDB EE                <1>       out    dx, al
9756 0000FEDC 88E8              <1>       mov    al, ch ; Blue
9757 0000FEDE EE                <1>       out    dx, al
9758 0000FEDF C1C108            <1>       rol    ecx, 8
9759 0000FEE2 88E1              <1>       mov    cl, ah ; Red
9760                            <1>              ; ecx = 00BBGGRRh
9761 0000FEE4 EBA8              <1>       jmp    short sysvideo_21_0
9762                            <1>
9763                            <1> sysvideo_21_5:
9764                            <1>       ; write/set all DAC color registers
9765 0000FEE6 89CE              <1>       mov    esi, ecx ; user's buffer address
9766 0000FEE8 BF00600900        <1>       mov    edi, VBE3STACKADDR
9767 0000FEED 89FB              <1>       mov    ebx, edi
9768 0000FEEF B900030000        <1>       mov    ecx, 768 ; 256*3
9769 0000FEF4 E815140000        <1>       call   transfer_from_user_buffer
9770 0000FEF9 7299              <1>       jc     short sysvideo_21_1
9771 0000FEFB 890D[64030300]    <1>       mov    [u.r0], ecx ; actual transfer count
9772                            <1>
9773 0000FF01 89DE              <1>       mov    esi, ebx ; VBE3STACKADDR
9774 0000FF03 66BAC803          <1>       mov    dx, 3C8h ; VGAREG_DAC_WRITE_ADDRESS
9775 0000FF07 28C0              <1>       sub    al, al ; 0
9776 0000FF09 EE                <1>       out    dx, al
9777                            <1>       ;mov   dx, 3C9h ; VGAREG_DAC_DATA
9778 0000FF0A FEC2              <1>       inc    dl
9779                            <1> sysvideo_21_6:
9780 0000FF0C AC                <1>       lodsb
9781 0000FF0D EE                <1>       out    dx, al
9782 0000FF0E AC                <1>       lodsb
9783 0000FF0F EE                <1>       out    dx, al
9784 0000FF10 AC                <1>       lodsb
9785 0000FF11 EE                <1>       out    dx, al
9786 0000FF12 E2F8              <1>       loop   sysvideo_21_6
9787 0000FF14 EB33              <1>       jmp    short sysvideo_21_9
9788                            <1>
9789                            <1> sysvideo_21_7:
9790                            <1>       ; BMP file type palette handling
9791                            <1>
9792 0000FF16 F6C301            <1>       test   bl, 1
9793 0000FF19 7571              <1>       jnz    short sysvideo_21_12 ; set/write DAC colors
9794                            <1>
9795                            <1>       ; Read DAC color register or all DAC color registers
9796 0000FF1B F6C302            <1>       test   bl, 2  ; read single DAC color register
9797 0000FF1E 742E              <1>       jz     short sysvideo_21_10 ; read all DAC color regs
9798                            <1>
9799                            <1>       ; read single DAC color register
9800                            <1>       ; CL = DAC color register (index)
9801                            <1>
9802 0000FF20 66BAC703          <1>       mov    dx, 3C7h ; VGAREG_DAC_READ_ADDRESS
9803 0000FF24 88C8              <1>       mov    al, cl ; DAC color register
9804 0000FF26 31C9              <1>       xor    ecx, ecx
9805 0000FF28 EE                <1>       out    dx, al
9806                            <1>       ;mov   dx, 3C9h ; VGAREG_DAC_DATA
9807 0000FF29 B2C9              <1>       mov    dl, 0C9h
9808 0000FF2B EC                <1>       in     al, dx
9809 0000FF2C C0E002            <1>       shl    al, 2
9810 0000FF2F 88C5              <1>       mov    ch, al ; red
9811 0000FF31 EC                <1>       in     al, dx
9812 0000FF32 C0E002            <1>       shl    al, 2
9813 0000FF35 88C1              <1>       mov    cl, al ; green
9814 0000FF37 EC                <1>       in     al, dx
9815 0000FF38 C1E108            <1>       shl    ecx, 8
9816 0000FF3B C0E002            <1>       shl    al, 2
9817 0000FF3E 88C1              <1>       mov    cl, al ; blue
9818 0000FF40 C1C910            <1>       ror    ecx, 16
9819                            <1>       ; CL = Blue, CH = Green, byte 3 = Red, byte 4 = 0
9820                            <1> sysvideo_21_8:
9821 0000FF43 890D[64030300]    <1>       mov    [u.r0], ecx
9822                            <1> sysvideo_21_9:
9823 0000FF49 E99ED9FFFF        <1>       jmp    sysret
9824                            <1> sysvideo_21_10:
```

```
9825                                    <1>       ; read all DAC color registers
9826 0000FF4E 89CD                      <1>       mov   ebp, ecx ; user's buffer address
9827 0000FF50 BF00600900                <1>       mov   edi, VBE3STACKADDR
9828 0000FF55 89FE                      <1>       mov   esi, edi
9829 0000FF57 B900040000                <1>       mov   ecx, 1024 ; 256*4
9830 0000FF5C 51                        <1>       push  ecx
9831 0000FF5D 66BAC703                  <1>       mov   dx, 3C7h ; VGAREG_DAC_READ_ADDRESS
9832 0000FF61 28C0                      <1>       sub   al, al ; 0
9833 0000FF63 EE                        <1>       out   dx, al
9834                                    <1>       ;mov  dx, 3C9h ; VGAREG_DAC_DATA
9835 0000FF64 B2C9                      <1>       mov   dl, 0C9h
9836                                    <1> sysvideo_21_11:
9837 0000FF66 31DB                      <1>       xor   ebx, ebx
9838 0000FF68 EC                        <1>       in    al, dx ; Red
9839 0000FF69 C0E002                    <1>       shl   al, 2
9840 0000FF6C 88C7                      <1>       mov   bh, al
9841 0000FF6E EC                        <1>       in    al, dx ; Green
9842 0000FF6F C0E002                    <1>       shl   al, 2
9843 0000FF72 88C3                      <1>       mov   bl, al
9844 0000FF74 EC                        <1>       in    al, dx ; Blue
9845 0000FF75 C0E002                    <1>       shl   al, 2
9846 0000FF78 C1E308                    <1>       shl   ebx, 8
9847 0000FF7B 89D8                      <1>       mov   eax, ebx ; 00RRGGBBh
9848 0000FF7D AB                        <1>       stosd
9849 0000FF7E E2E6                      <1>       loop  sysvideo_21_11
9850 0000FF80 59                        <1>       pop   ecx
9851                                    <1>
9852 0000FF81 89EF                      <1>       mov   edi, ebp ; user's buffer address
9853                                    <1>       ;mov  esi, VBE3STACKADDR
9854                                    <1>       ;mov  ecx, 1024 = 4*256
9855 0000FF83 E83C130000                <1>       call  transfer_to_user_buffer
9856 0000FF88 72BF                      <1>       jc    short sysvideo_21_9
9857                                    <1>       ;mov  [u.r0], ecx ; actual transfer count
9858 0000FF8A EBB7                      <1>       jmp   short sysvideo_21_8
9859                                    <1>
9860                                    <1> sysvideo_21_12:
9861                                    <1>       ; Set/Write DAC color register or all registers
9862 0000FF8C F6C302                    <1>       test  bl, 2 ; write/set single DAC color register
9863 0000FF8F 7427                      <1>       jz    short sysvideo_21_13 ; set all DAC color regs
9864                                    <1>
9865                                    <1>       ; set single DAC color register
9866                                    <1>       ; CL = DAC color register (index)
9867                                    <1>       ; (byte 1 = Blue, byte 2 = Green, byte 3 = Red)
9868                                    <1>
9869 0000FF91 66BAC803                  <1>       mov   dx, 3C8h ; VGAREG_DAC_WRITE_ADDRESS
9870 0000FF95 88C8                      <1>       mov   al, cl ; DAC color register (index)
9871 0000FF97 88EC                      <1>       mov   ah, ch ; Blue
9872 0000FF99 C1E910                    <1>       shr   ecx, 16
9873 0000FF9C EE                        <1>       out   dx, al
9874                                    <1>       ;mov  dx, 3C9h ; VGAREG_DAC_DATA
9875 0000FF9D FEC2                      <1>       inc   dl
9876 0000FF9F 88E8                      <1>       mov   al, ch ; Red
9877 0000FFA1 C0E802                    <1>       shr   al, 2
9878 0000FFA4 EE                        <1>       out   dx, al
9879 0000FFA5 88C8                      <1>       mov   al, cl ; Green
9880 0000FFA7 C0E802                    <1>       shr   al, 2
9881 0000FFAA EE                        <1>       out   dx, al
9882 0000FFAB 88E0                      <1>       mov   al, ah ; Blue
9883 0000FFAD C0E802                    <1>       shr   al, 2
9884 0000FFB0 EE                        <1>       out   dx, al
9885 0000FFB1 C1C108                    <1>       rol   ecx, 8
9886 0000FFB4 88E1                      <1>       mov   cl, ah
9887 0000FFB6 EB8B                      <1>       jmp   short sysvideo_21_8
9888                                    <1>
9889                                    <1> sysvideo_21_13:
9890                                    <1>       ; write/set all DAC color registers
9891 0000FFB8 89CE                      <1>       mov   esi, ecx ; user's buffer address
9892 0000FFBA BF00600900                <1>       mov   edi, VBE3STACKADDR
9893 0000FFBF 89FB                      <1>       mov   ebx, edi
9894 0000FFC1 B900040000                <1>       mov   ecx, 1024 ; 256*4
9895 0000FFC6 E843130000                <1>       call  transfer_from_user_buffer
9896 0000FFCB 7229                      <1>       jc    short sysvideo_21_15
9897 0000FFCD 890D[64030300]            <1>       mov   [u.r0], ecx ; actual transfer count
9898                                    <1>
9899 0000FFD3 89DE                      <1>       mov   esi, ebx ; VBE3STACKADDR
9900 0000FFD5 66BAC803                  <1>       mov   dx, 3C8h ; VGAREG_DAC_WRITE_ADDRESS
9901 0000FFD9 28C0                      <1>       sub   al, al ; 0
9902 0000FFDB EE                        <1>       out   dx, al
9903                                    <1>       ;mov  dx, 3C9h ; VGAREG_DAC_DATA
9904 0000FFDC FEC2                      <1>       inc   dl
9905                                    <1> sysvideo_21_14:
9906 0000FFDE AD                        <1>       lodsd
9907                                    <1>       ; byte 0 = Blue, byte 1 = Green, byte 2 = Red
9908 0000FFDF 89C1                      <1>       mov   ecx, eax
9909 0000FFE1 C1E010                    <1>       shl   eax, 16 ; byte 0 = Red
9910 0000FFE4 C0E802                    <1>       shr   al, 2
9911 0000FFE7 EE                        <1>       out   dx, al ; Red
9912 0000FFE8 88E8                      <1>       mov   al, ch
9913 0000FFEA C0E802                    <1>       shr   al, 2
9914 0000FFED EE                        <1>       out   dx, al ; Green
9915 0000FFEE 88C8                      <1>       mov   al, cl
9916 0000FFF0 C0E802                    <1>       shr   al, 2
9917 0000FFF3 EE                        <1>       out   dx, al ; Blue
9918 0000FFF4 E2E8                      <1>       loop  sysvideo_21_14
9919                                    <1> sysvideo_21_15:
9920 0000FFF6 E9F1D8FFFF                <1>       jmp   sysret
9921                                    <1>
9922                                    <1> sysvideo_22:
9923                                    <1>       ; 22/01/2021
9924                                    <1>       ; 17/01/2021
9925                                    <1>       ; 04/12/2020
9926                                    <1>       ; 03/12/2020
9927                                    <1>       ; BH = 9
9928                                    <1>       ; Set/Get VESA VBE3 protected mode interface params
9929                                    <1>
```

```
9930                                    <1>       ; 22/01/2021
9931                                    <1>       ;cmp   byte [vbe3], 3
9932                                    <1>       ;jne   short sysvideo_25 ; not applicable if
9933                                    <1>                                ; vbe3 compatible video bios
9934                                    <1>                                ; is not detected by kernel
9935 0000FFFB 80FB02                   <1>       cmp    bl, 2
9936                                    <1>       ;ja    short sysvideo_25 ; bl > 2 not implemented
9937                                    <1>       ; 17/01/2021
9938 0000FFFE 7716                     <1>       ja     short sysvideo_22_0 ; srvs flag sub function
9939                                    <1>       ;jb    short sysvideo_23
9940                                    <1>
9941                                    <1>       ; 21/01/2021
9942 00010000 803D[54090000]03         <1>       cmp    byte [vbe3], 3
9943 00010007 757D                     <1>       jne    short sysvideo_25 ; not applicable if
9944                                    <1>                                ; vbe3 compatible video bios
9945                                    <1>                                ; is not detected by kernel
9946 00010009 80FB01                   <1>       cmp    bl, 1
9947 0001000C 7663                     <1>       jna    short sysvideo_23
9948                                    <1>
9949 0001000E 8A1D[1C120300]           <1>       mov    bl, [pmi32] ; Video bios 32 bit PMI functions
9950 00010014 EB68                     <1>       jmp    short sysvideo_24
9951                                    <1>
9952                                    <1> sysvideo_22_0:
9953                                    <1>       ; 17/01/2021
9954                                    <1>       ; save/restore video state user permission
9955 00010016 80FB05                   <1>       cmp    bl, 5
9956 00010019 771E                     <1>       ja     short sysvideo_22_2
9957 0001001B 7208                     <1>       jb     short sysvideo_22_1
9958                                    <1>       ; get srvs flag value/status
9959 0001001D 8A1D[84120300]           <1>       mov    bl, [srvsf] ; 0 = disabled, 1 = enabled
9960 00010023 EB2C                     <1>       jmp    short sysvideo_22_3
9961                                    <1>
9962                                    <1> sysvideo_22_1:
9963                                    <1>       ; permission (root and multi tasking) check
9964 00010025 E836000000               <1>       call   sysvideo_22_4
9965 0001002A 735A                     <1>       jnc    short sysvideo_25 ; not permitted !
9966                                    <1>       ; cf = 1
9967 0001002C 80EB03                   <1>       sub    bl, 3 ; disable = 0, enable = 1
9968                                    <1>       ; 22/01/2021
9969 0001002F 881D[84120300]           <1>       mov    [srvsf], bl
9970 00010035 FEC3                     <1>       inc    bl ; 1 = disabled, 2 = enabled
9971 00010037 EB18                     <1>       jmp    short sysvideo_22_3
9972                                    <1>
9973                                    <1> sysvideo_22_2:
9974 00010039 80FB06                   <1>       cmp    bl, 6
9975 0001003C 7748                     <1>       ja     short sysvideo_25 ; invalid/unimplemented
9976                                    <1>       ; get VESA VBE number/status
9977 0001003E 8A25[54090000]           <1>       mov    ah, [vbe3] ; vbe3 = 3, vbe2 = 2, others = 0
9978 00010044 A0[55090000]             <1>       mov    al, [vbe2bios] ; bochs/qemu/vbox emulator status
9979 00010049 66A3[64030300]           <1>       mov    [u.r0], ax
9980 0001004F EB35                     <1>       jmp    short sysvideo_25
9981                                    <1>
9982                                    <1> sysvideo_22_3:
9983                                    <1>       ; 22/01/2021
9984 00010051 8A3D[85120300]           <1>       mov    bh, [srvso] ; state options (> 80h -> svga)
9985 00010057 66891D[64030300]         <1>       mov    [u.r0], bx ; function result is return value
9986 0001005E EB26                     <1>       jmp    short sysvideo_25
9987                                    <1>
9988                                    <1> sysvideo_22_4:
9989                                    <1>       ; 17/01/2021 - permission will be given by root only
9990 00010060 803D[228E0100]00         <1>       cmp    byte [multi_tasking], 0 ; in single user mode
9991 00010067 7707                     <1>       ja     short sysvideo_22_5
9992                                    <1>       ; 19/01/2021
9993 00010069 803D[B0030300]01         <1>       cmp    byte [u.uid], 1 ; ([u.uid] = 0 -> root)
9994                                    <1> sysvideo_22_5:
9995                                    <1>       ; [multi_tasking] = 0 & [u.uid] = 0  -> CF = 1
9996                                    <1>       ; otherwise -> CF = 0
9997 00010070 C3                       <1>       retn
9998                                    <1>
9999                                    <1> sysvideo_23:
10000                                   <1>       ; 17/01/2021
10001                                   <1>       ; permission (root and multi tasking) check
10002 00010071 E8EAFFFFFF               <1>       call   sysvideo_22_4
10003 00010076 730E                     <1>       jnc    short sysvideo_25 ; not permitted !
10004                                   <1>
10005 00010078 881D[1C120300]           <1>       mov    [pmi32], bl ; 1 = enabled, 0 = disabled
10006                                   <1> sysvideo_24:
10007 0001007E FEC3                     <1>       inc    bl
10008 00010080 881D[64030300]           <1>       mov    [u.r0], bl ; function result is return value
10009                                   <1> sysvideo_25:
10010 00010086 E961D8FFFF               <1>       jmp    sysret
10011                                   <1>
10012                                   <1> sysvideo_26:
10013                                   <1>       ; 23/12/2020
10014                                   <1>       ; BH = 10
10015                                   <1>       ; Map video memory to user's buffer
10016                                   <1>       ; (multiuser/owner r/w permisions are ignored
10017                                   <1>       ;  for current TRDOS 386 version !)
10018                                   <1>
10019 0001008B 6681E100F0              <1>       and    cx, ~4095  ; clear low 12 bits
10020 00010090 09C9                    <1>       or     ecx, ecx ; start address of user's buffer
10021 00010092 74F2                    <1>       jz     short sysvideo_25 ; error !
10022                                   <1>
10023 00010094 80FB01                  <1>       cmp    bl, 1  ; VGA memory mapping ?
10024 00010097 740E                    <1>       je     short sysvideo_26_1
10025 00010099 7718                    <1>       ja     short sysvideo_26_2
10026                                   <1> sysvideo_26_0:
10027                                   <1>       ; BL = 0 : CGA memory (0B8000h) map (32K)
10028 0001009B B800800B00              <1>       mov    eax, 0B8000h
10029 000100A0 BB00800000              <1>       mov    ebx, 32768
10030 000100A5 EB37                    <1>       jmp    short sysvideo_26_3
10031                                   <1> sysvideo_26_1:
10032                                   <1>       ; BL = 1 : VGA memory (0A0000h) map (64K)
10033 000100A7 B800000A00              <1>       mov    eax, 0A0000h
10034 000100AC BB00000100              <1>       mov    ebx, 65536
```

```
10035 000100B1 EB2B                <1>       jmp    short sysvideo_26_3
10036                              <1> sysvideo_26_2:
10037                              <1>       ; BL = 2 : SVGA memory (LFB) map to user's buffer
10038 000100B3 803D[54090000]02    <1>       cmp    byte [vbe3], 2  ; VESA VBE 2/3 vbios ready ?
10039 000100BA 72CA                <1>       jb     short sysvideo_25  ; no, error !
10040 000100BC 6681E200F0          <1>       and    dx, ~4095 ; clear low 12 bits
10041 000100C1 09D2                <1>       or     edx, edx  ; buffer size in bytes
10042 000100C3 74C1                <1>       jz     short sysvideo_25  ; error
10043 000100C5 89D3                <1>       mov    ebx, edx
10044 000100C7 A1[2C120300]        <1>       mov    eax, [LFB_ADDR] ; [LFB_Info+LFBINFO.LFB_addr]
10045 000100CC 21C0                <1>       and    eax, eax
10046 000100CE 7425                <1>       jz     short sysvideo_26_5
10047                              <1>                           ; (LFB parms are not set yet)
10048 000100D0 3B1D[30120300]      <1>       cmp    ebx, [LFB_SIZE] ; [LFB_Info+LFBINFO.LFB_size]
10049 000100D6 7606                <1>       jna    short sysvideo_26_3
10050 000100D8 8B1D[30120300]      <1>       mov    ebx, [LFB_SIZE]
10051                              <1> sysvideo_26_3:
10052 000100DE 52                  <1>       push   edx
10053 000100DF 53                  <1>       push   ebx    ; buffer size in bytes
10054 000100E0 51                  <1>       push   ecx    ; user's buffer address
10055 000100E1 87D9                <1>       xchg   ebx, ecx
10056 000100E3 C1E90C              <1>       shr    ecx, 12 ; convert buffer size to page count
10057 000100E6 E80566FFFF          <1>       call   direct_memory_access
10058 000100EB 59                  <1>       pop    ecx    ; user's buffer address
10059 000100EC 5B                  <1>       pop    ebx    ; buffer size
10060 000100ED 5A                  <1>       pop    edx
10061 000100EE 7296                <1>       jc     short sysvideo_25 ; error !
10062                              <1>                            ; [u.r0] = 0
10063                              <1> ;sysvideo_26_4:
10064                              <1>       ;mov   ebp, [u.usp] ; ebp points to user's registers
10065                              <1>       ;mov   [ebp+20], edx ; return to user with EDX value
10066                              <1>       ;mov   [ebp+16], ebx ; EBX
10067                              <1>       ;mov   [ebp+24], ecx ; ECX
10068                              <1>       ; eax = physical address of video memory (LFB)
10069                              <1>       ;mov   [u.r0], eax
10070                              <1>       ;jmp   sysret
10071 000100F0 E946FCFFFF          <1>       jmp    sysvideo_26_4
10072                              <1>
10073                              <1> sysvideo_26_5:
10074 000100F5 66A1[E30E0000]      <1>       mov    ax, [def_LFB_addr] ; default LFB for mode 118h
10075                              <1>       ; ah must be 0C0h or 0D0h or E0h
10076                              <1>       ; others are nonsence !?
10077 000100FB 08E4                <1>       or     ah, ah
10078 000100FD 7487                <1>       jz     short sysvideo_25 ; invalid lfb addr or
10079                              <1>                           ; it is not a vbe2 -bochs emu-
10080                              <1>                           ; or vbe3 -real- video bios
10081 000100FF 80FCF0              <1>       cmp    ah, 0F0h
10082 00010102 7382                <1>       jnb    short sysvideo_25 ; nonsence !?
10083 00010104 C1E010              <1>       shl    eax, 16
10084                              <1>       ;jz    short sysvideo_25 ; eax = 0
10085                              <1>
10086 00010107 81FB00907E00        <1>       cmp    ebx, 1920*1080*4 ; maximum value of possible
10087                              <1>                             ; buffer sizes
10088 0001010D 76CF                <1>       jna    short sysvideo_26_3  ; buffer size is proper
10089                              <1>       ; resize buffer to fit 4GB limit
10090 0001010F BB00907E00          <1>       mov    ebx, 1920*1080*4
10091 00010114 EBC8                <1>       jmp    short sysvideo_26_3
10092                              <1>
10093                              <1> sysvideo_27:
10094                              <1>       ; 18/01/2021
10095 00010116 80FF0C              <1>       cmp    bh, 12
10096 00010119 0F877E010000        <1>       ja     sysvideo_28 ; 19/01/2021
10097                              <1>
10098                              <1>       ; BH = 12
10099                              <1>       ; Font sub functions.
10100                              <1>       ; 11/01/2021
10101                              <1>       ; 10/01/2021
10102                              <1>       ;   BL = 0 : Disable system font overwrite
10103                              <1>       ;   BL = 1 : Enable system font overwrite
10104                              <1>       ;   BL = 2 : Read system font 8x8
10105                              <1>       ;   BL = 3 : Read system font 8x14
10106                              <1>       ;   BL = 4 : Read system font 8x16
10107                              <1>       ;   BL = 5 : Read user defined font 8x8
10108                              <1>       ;   BL = 6 : Read user defined font 8x16
10109                              <1>       ;   BL = 7 : Write system font 8x8
10110                              <1>       ;   BL = 8 : Write system font 8x14
10111                              <1>       ;   BL = 9 : Write system font 8x16
10112                              <1>       ;   BL = 10 : Write user defined font 8x8
10113                              <1>       ;   BL = 11 : Write user defined font 8x16
10114                              <1>       ;
10115                              <1>       ;  BL > 11 : invalid (not implemented)
10116                              <1>       ;
10117                              <1>       ; For BL = 1 to 11
10118                              <1>       ;   ECX = number of characters (>= 256)
10119                              <1>       ;   EDX = first character (ascii code in DL)
10120                              <1>       ;   ESI = user's buffer address
10121                              <1>       ;
10122                              <1>       ; Return: EAX = character count
10123                              <1>
10124 0001011F 80FB0B              <1>       cmp    bl, 11
10125 00010122 7605                <1>       jna    short sysvideo_27_1
10126                              <1> sysvideo_27_0:
10127 00010124 E9C3D7FFFF          <1>       jmp    sysret ; not implemented yet !
10128                              <1> sysvideo_27_1:
10129 00010129 66B80001            <1>       mov    ax, 256
10130 0001012D 08DB                <1>       or     bl, bl
10131 0001012F 750E                <1>       jnz    short sysvideo_27_3
10132                              <1>
10133                              <1>       ; bl = 0
10134                              <1>       ; enable system font overwrite
10135                              <1>
10136 00010131 8025[82120300]7F    <1>       and    byte [ufont], 7Fh ; clear bit 7
10137                              <1> sysvideo_27_2:
10138                              <1>       ;mov   word [u.r0], 256  ; > 0 -> successful
10139 00010138 A3[64030300]        <1>       mov    [u.r0], eax ; 256
```

```
10140 0001013D EBE5               <1>       jmp    short sysvideo_27_0
10141                             <1> sysvideo_27_3:
10142 0001013F 80FB01             <1>       cmp    bl, 1
10143 00010142 7710               <1>       ja     short sysvideo_27_4
10144                             <1>
10145                             <1>       ; bl = 1
10146                             <1>       ; enable system font overwrite
10147                             <1>       ;      if [multi_tasking]= 0 and [u.uid] = 0
10148                             <1>
10149                             <1>       ;cmp   byte [multi_tasking], 0
10150                             <1>       ;                   ; multi tasking enabled ?
10151                             <1>       ;ja    short sysvideo_27_0 ; yes
10152                             <1>       ;; 19/01/2021
10153                             <1>       ;; system maintenance or single user mode
10154                             <1>       ;cmp   byte [u.uid], 0  ; root ?
10155                             <1>       ;ja    short sysvideo_27_0 ; no
10156                             <1>
10157                             <1>       ; 19/01/2021
10158                             <1>       ; multi tasking & root check
10159 00010144 E817FFFFFF        <1>       call   sysvideo_22_4
10160 00010149 73D9               <1>       jnc    short sysvideo_27_0 ; not permitted
10161                             <1>
10162                             <1>       ; [multi_tasking]= 0 and [u.uid] = 0
10163                             <1>
10164 0001014B 800D[82120300]80   <1>       or     byte [ufont], 80h ; set bit 7
10165                             <1>
10166 00010152 EBE4               <1>       jmp    short sysvideo_27_2
10167                             <1>
10168                             <1> sysvideo_27_4:
10169 00010154 09C9               <1>       or     ecx, ecx
10170 00010156 74CC               <1>       jz     short sysvideo_27_0
10171 00010158 21D2               <1>       and    edx, edx
10172 0001015A 7410               <1>       jz     short sysvideo_27_4_0
10173                             <1>       ;mov   ax, 256
10174 0001015C 39C1               <1>       cmp    ecx, eax ; 256
10175 0001015E 77C4               <1>       ja     short sysvideo_27_0
10176 00010160 48                 <1>       dec    eax
10177 00010161 39C2               <1>       cmp    edx, eax ; 255
10178 00010163 77BF               <1>       ja     short sysvideo_27_0
10179 00010165 40                 <1>       inc    eax
10180 00010166 29D0               <1>       sub    eax, edx ; 256 - DX
10181 00010168 39C8               <1>       cmp    eax, ecx
10182 0001016A 72B8               <1>       jb     short sysvideo_27_0
10183                             <1>
10184                             <1> sysvideo_27_4_0:
10185 0001016C 89F5               <1>       mov    ebp, esi
10186                             <1>
10187 0001016E 80FB06             <1>       cmp    bl, 6
10188 00010171 776E               <1>       ja     short sysvideo_27_13
10189 00010173 7210               <1>       jb     short sysvideo_27_5
10190                             <1>       ; bl = 6
10191 00010175 F605[82120300]10   <1>       test   byte [ufont], 16 ; 8x16 user font loaded ?
10192 0001017C 74A6               <1>       jz     short sysvideo_27_0
10193                             <1>       ; read 8x16 user defined font
10194 0001017E BE00400900         <1>       mov    esi, VGAFONT16USER
10195 00010183 EB0C               <1>       jmp    short sysvideo_27_6
10196                             <1> sysvideo_27_5:
10197 00010185 80FB04             <1>       cmp    bl, 4
10198 00010188 723F               <1>       jb     short sysvideo_27_11
10199 0001018A 7725               <1>       ja     short sysvideo_27_9
10200                             <1>       ; bl = 4
10201                             <1>       ; read 8x16 system font
10202 0001018C BE[246E0100]       <1>       mov    esi, vgafont16
10203                             <1> sysvideo_27_6:
10204                             <1>       ; read 8x16 font
10205 00010191 66C1E204           <1>       shl    dx, 4 ; * 16
10206 00010195 66C1E104           <1>       shl    cx, 4 ; * 16 ; 16 bytes per char
10207                             <1> sysvideo_27_7:
10208 00010199 89EF               <1>       mov    edi, ebp
10209 0001019B 01D7               <1>       add    edi, edx
10210 0001019D 01D6               <1>       add    esi, edx
10211                             <1>       ; ecx = byte count
10212                             <1>       ; esi = source (in system memory)
10213                             <1>       ; edi = destination (in user memory)
10214 0001019F E820110000         <1>       call   transfer_to_user_buffer
10215 000101A4 7206               <1>       jc     short sysvideo_27_8
10216 000101A6 890D[64030300]     <1>       mov    [u.r0], ecx
10217                             <1> sysvideo_27_8:
10218 000101AC E93BD7FFFF         <1>       jmp    sysret
10219                             <1> sysvideo_27_9:
10220                             <1>       ; bl = 5
10221 000101B1 F605[82120300]08   <1>       test   byte [ufont], 8 ; 8x8 user font loaded ?
10222 000101B8 74F2               <1>       jz     short sysvideo_27_8
10223                             <1>       ; read 8x8 user defined font
10224 000101BA BE00500900         <1>       mov    esi, VGAFONT8USER
10225                             <1> sysvideo_27_10:
10226                             <1>       ; read 8x8 font
10227 000101BF 66C1E203           <1>       shl    dx, 3 ; * 8
10228 000101C3 66C1E103           <1>       shl    cx, 3 ; * 8 ; 8 bytes per char
10229 000101C7 EBD0               <1>       jmp    short sysvideo_27_7
10230                             <1>
10231                             <1> sysvideo_27_11:
10232 000101C9 80FB03             <1>       cmp    bl, 3  ; 8x14 system font
10233 000101CC 720C               <1>       jb     short sysvideo_27_12 ; 8x8 system font
10234                             <1>       ; bl = 3
10235                             <1>       ; read 8x14 system font
10236                             <1>       ;mov   al, 14
10237                             <1>       ;mul   dl
10238                             <1>       ;mov   dx, ax
10239                             <1>       ;push  edx
10240                             <1>       ;mov   ax, 14
10241                             <1>       ;mul   cx
10242                             <1>       ;mov   cx, ax
10243                             <1>       ;pop   edx
10244 000101CE E8AB000000         <1>       call   sysvideo_27_14
```

```
10245 000101D3 BE[24600100]        <1>       mov   esi, vgafont14
10246 000101D8 EBBF                <1>       jmp   short sysvideo_27_7
10247                              <1>
10248                              <1> sysvideo_27_12:
10249                              <1>       ; bl = 2
10250                              <1>       ; read 8x8 system font
10251 000101DA BE[24580100]        <1>       mov   esi, vgafont8
10252 000101DF EBDE                <1>       jmp   short sysvideo_27_10
10253                              <1>
10254                              <1> sysvideo_27_13:
10255                              <1>       ; overwrite font
10256 000101E1 80FB0A              <1>       cmp   bl, 10
10257 000101E4 7774                <1>       ja    short sysvideo_27_22 ; 8x16 user font
10258 000101E6 7224                <1>       jb    short sysvideo_27_15
10259                              <1>       ; bl = 10
10260 000101E8 BF00500900          <1>       mov   edi, VGAFONT8USER
10261 000101ED F605[82120300]08    <1>       test  byte [ufont], 8 ; 8x8 user font loaded ?
10262 000101F4 755A                <1>       jnz   short sysvideo_27_21 ; yes
10263 000101F6 08ED                <1>       or    ch, ch ; cx = 256
10264                              <1>       ;jnz  short sysvideo_27_21 ; 256 chars
10265 000101F8 7406                <1>       jz    short sysvideo_27_13_0
10266 000101FA 66B90008            <1>       mov   cx, 8*256
10267 000101FE EB39                <1>       jmp   short sysvideo_27_18_0
10268                              <1> sysvideo_27_13_0:
10269                              <1>       ; copy system font to user font before overwrite
10270 00010200 BE[24580100]        <1>       mov   esi, vgafont8
10271                              <1>       ;push edi
10272                              <1>       ;push ecx
10273                              <1>       ;mov  cl, 64
10274                              <1>       ;rep  movsd
10275                              <1>       ;pop  ecx
10276                              <1>       ;pop  edi
10277                              <1>       ;mov  esi, ebp ; user's font buffer
10278 00010205 E888000000          <1>       call  sysvideo_27_23
10279 0001020A EB44                <1>       jmp   short sysvideo_27_21
10280                              <1>
10281                              <1> sysvideo_27_15:
10282                              <1>       ; check system font overwrite permission
10283 0001020C F605[82120300]80    <1>       test  byte [ufont], 80h
10284 00010213 7497                <1>       jz    short sysvideo_27_8
10285                              <1>
10286 00010215 80FB08              <1>       cmp   bl, 8
10287 00010218 7740                <1>       ja    short sysvideo_27_22 ; 8x16 system font
10288 0001021A 722F                <1>       jb    short sysvideo_27_20 ; 8x8 system font
10289                              <1>       ; bl = 8
10290                              <1>       ; overwrite 8x14 system font
10291                              <1>       ;mov  al, 14
10292                              <1>       ;mul  dl
10293                              <1>       ;mov  dx, ax
10294                              <1>       ;push edx
10295                              <1>       ;mov  ax, 14
10296                              <1>       ;mul  cx
10297                              <1>       ;mov  cx, ax
10298                              <1>       ;pop  edx
10299 0001021C E85D000000          <1>       call  sysvideo_27_14
10300 00010221 BF[24600100]        <1>       mov   edi, vgafont14
10301 00010226 EB0D                <1>       jmp   short sysvideo_27_18
10302                              <1> sysvideo_27_16:
10303                              <1>       ; bl = 9
10304                              <1>       ; overwrite 8x16 system font
10305 00010228 BF[246E0100]        <1>       mov   edi, vgafont16
10306                              <1> sysvideo_27_17:
10307                              <1>       ; overwrite 8x16 font
10308 0001022D 66C1E204            <1>       shl   dx, 4 ; * 16
10309 00010231 66C1E104            <1>       shl   cx, 4 ; * 16 ; 16 bytes per char
10310                              <1> sysvideo_27_18:
10311 00010235 01D7                <1>       add   edi, edx
10312 00010237 01D6                <1>       add   esi, edx
10313                              <1> sysvideo_27_18_0:
10314                              <1>       ; ecx = byte count
10315                              <1>       ; esi = source (in user memory)
10316                              <1>       ; edi = destination (in system memory)
10317 00010239 E8D0100000          <1>       call  transfer_from_user_buffer
10318 0001023E 7206                <1>       jc    short sysvideo_27_19
10319 00010240 890D[64030300]      <1>       mov   [u.r0], ecx
10320                              <1> sysvideo_27_19:
10321 00010246 E9A1D6FFFF          <1>       jmp   sysret
10322                              <1> sysvideo_27_20:
10323                              <1>       ; bl = 7
10324                              <1>       ; overwrite 8x8 system font
10325 0001024B BF[24580100]        <1>       mov   edi, vgafont8
10326                              <1> sysvideo_27_21:
10327                              <1>       ; write 8x8 font
10328 00010250 66C1E203            <1>       shl   dx, 3 ; * 8
10329 00010254 66C1E103            <1>       shl   cx, 3 ; * 8 ; 8 bytes per char
10330 00010258 EBDB                <1>       jmp   short sysvideo_27_18
10331                              <1> sysvideo_27_22:
10332                              <1>       ; bl = 11
10333                              <1>       ; overwrite 8x16 user defined font
10334 0001025A BF00400900          <1>       mov   edi, VGAFONT16USER
10335 0001025F F605[82120300]10    <1>       test  byte [ufont], 16 ; 8x16 user font loaded ?
10336 00010266 75C5                <1>       jnz   short sysvideo_27_17 ; yes
10337 00010268 08ED                <1>       or    ch, ch ; cx = 256
10338                              <1>       ;jnz  short sysvideo_27_17 ; 256 chars
10339 0001026A 7406                <1>       jz    short sysvideo_27_22_0
10340 0001026C 66B90010            <1>       mov   cx, 16*256
10341 00010270 EBC7                <1>       jmp   short sysvideo_27_18_0
10342                              <1> sysvideo_27_22_0:
10343                              <1>       ; copy system font to user font before overwrite
10344 00010272 BE[246E0100]        <1>       mov   esi, vgafont16
10345                              <1>       ;push edi
10346                              <1>       ;push ecx
10347                              <1>       ;mov  cl, 64
10348                              <1>       ;rep  movsd
10349                              <1>       ;pop  ecx
```

```
10350                                   <1>       ;pop   edi
10351                                   <1>       ;mov   esi, ebp ; user's font buffer
10352 00010277 E816000000              <1>       call   sysvideo_27_23
10353 0001027C EBAF                    <1>       jmp    short sysvideo_27_17
10354                                   <1>
10355                                   <1> sysvideo_27_14:
10356 0001027E B00E                    <1>       mov    al, 14
10357 00010280 F6E2                    <1>       mul    dl
10358 00010282 6689C2                  <1>       mov    dx, ax
10359 00010285 52                      <1>       push   edx
10360 00010286 66B80E00                <1>       mov    ax, 14
10361 0001028A 66F7E1                  <1>       mul    cx
10362 0001028D 6689C1                  <1>       mov    cx, ax
10363 00010290 5A                      <1>       pop    edx
10364 00010291 C3                      <1>       retn
10365                                   <1>
10366                                   <1> sysvideo_27_23:
10367 00010292 57                      <1>       push   edi
10368 00010293 51                      <1>       push   ecx
10369 00010294 B140                    <1>       mov    cl, 64
10370 00010296 F3A5                    <1>       rep    movsd
10371 00010298 59                      <1>       pop    ecx
10372 00010299 5F                      <1>       pop    edi
10373 0001029A 89EE                    <1>       mov    esi, ebp ; user's font buffer
10374 0001029C C3                      <1>       retn
10375                                   <1>
10376                                   <1> sysvideo_28:
10377                                   <1>       ; 24/01/2021
10378                                   <1>       ; 23/01/2021
10379                                   <1>       ; 18/01/2021
10380 0001029D 80FF0E                  <1>       cmp    bh, 14
10381 000102A0 0F8275010000            <1>       jb     sysvideo_29
10382 000102A6 0F8754020000            <1>       ja     sysvideo_30
10383                                   <1>
10384                                   <1>       ; BH = 14
10385                                   <1>       ; Save/Restore Super VGA video state
10386                                   <1>
10387                                   <1>       ; BL = options
10388                                   <1>       ;      bit 0 - Save (0) or Restore (1)
10389                                   <1>       ;      bit 1 - controller hardware state
10390                                   <1>       ;      bit 2 - BIOS data state
10391                                   <1>       ;      bit 3 - DAC state
10392                                   <1>       ;      bit 4 - (extended) Register state
10393                                   <1>       ;      bit 5 - system (0) or user (1) memory
10394                                   <1>       ;      bit 6 - verify without transfer
10395                                   <1>       ;      bit 7 - not used (must be 0)
10396                                   <1>
10397                                   <1>       ; ECX = Buffer address or VideoStateID
10398                                   <1>
10399 000102AC 803D[54090000]02        <1>       cmp    byte [vbe3], 2 ;  VESA VBE2 or VBE3 ?
10400 000102B3 7717                    <1>       ja     short sysvideo_28_0 ; yes
10401 000102B5 7210                    <1>       jb     short sysvideo_28_16 ; not a SVGA sys !
10402                                   <1>
10403                                   <1>       ; == VBE2 ==
10404                                   <1>       ; Check Bochs/Qemu/VirtualBox PC emulator
10405                                   <1>       ; (vbe2 is usable only for emulator's vbios)
10406 000102B7 8A25[55090000]          <1>       mov    ah, [vbe2bios]
10407 000102BD 80FCC0                  <1>       cmp    ah, 0C0h
10408 000102C0 7205                    <1>       jb     short sysvideo_28_16 ; unknown vbios !
10409 000102C2 80FCC5                  <1>       cmp    ah, 0C5h
10410 000102C5 7605                    <1>       jna    short sysvideo_28_0
10411                                   <1>              ; Use kernel's vbios functions (video.s)
10412                                   <1> sysvideo_28_16:
10413                                   <1>       ; unknown vbios !
10414 000102C7 E920D6FFFF              <1>       jmp    sysret
10415                                   <1>
10416                                   <1> sysvideo_28_0:
10417 000102CC 80FB7F                  <1>       cmp    bl, 7Fh
10418 000102CF 77F6                    <1>       ja     short sysvideo_28_16 ; unknown options
10419                                   <1>
10420 000102D1 88DA                    <1>       mov    dl, bl
10421 000102D3 80E21F                  <1>       and    dl, 1Fh
10422 000102D6 D0EA                    <1>       shr    dl, 1
10423 000102D8 74ED                    <1>       jz     short sysvideo_28_16 ; invalid !
10424                                   <1>       ; DL = VBE Function 4F04h Save/Restore options
10425                                   <1>       ;  bit 0 : controller hardware state
10426                                   <1>       ;  bit 1 : BIOS data state
10427                                   <1>       ;  bit 2 : DAC state
10428                                   <1>       ;  bit 3 : (extended) Register state
10429                                   <1>
10430 000102DA F6C320                  <1>       test   bl, 32 ; bit 5
10431 000102DD 0F85B1000000            <1>       jnz    sysvideo_28_7 ; user buffer
10432                                   <1>
10433                                   <1>       ; source or destination is kernel/system buffer
10434                                   <1>
10435 000102E3 803D[84120300]00        <1>       cmp    byte [srvsf], 0 ; srs permission flag
10436 000102EA 76DB                    <1>       jna    short sysvideo_28_16 ; not permitted
10437                                   <1>
10438 000102EC F6C301                  <1>       test   bl, 1
10439 000102EF 743A                    <1>       jz     short sysvideo_28_4 ; Save
10440                                   <1>
10441                                   <1>       ; Restore
10442 000102F1 3B0D[86120300]          <1>       cmp    ecx, [VideoStateID]
10443 000102F7 75CE                    <1>       jne    short sysvideo_28_16 ; not correct ID !
10444                                   <1>
10445 000102F9 0FB6CA                  <1>       movzx  ecx, dl
10446 000102FC 80CA80                  <1>       or     dl, 80h
10447 000102FF 3A15[85120300]          <1>       cmp    dl, [srvso]
10448 00010305 75C0                    <1>       jne    short sysvideo_28_16 ; not correct !
10449                                   <1>
10450 00010307 88DA                    <1>       mov    dl, bl
10451                                   <1>
10452                                   <1>       ; ecx = cl = options
10453 00010309 E85739FFFF              <1>       call   vbe_srs_gbs
10454                                   <1>       ; ebx = state buffer size (data size)
```

```
10455                           <1>
10456 0001030E 891D[64030300]   <1>       mov    [u.r0], ebx
10457                           <1>
10458 00010314 F6C240           <1>       test   dl, 64 ; verify without transfer
10459 00010317 75AE             <1>       jnz    short sysvideo_28_16 ; yes
10460                           <1>
10461 00010319 BE00580900       <1>       mov    esi, VBE3VIDEOSTATE
10462 0001031E BF00760900       <1>       mov    edi, VBE3SAVERESTOREBLOCK
10463 00010323 87CB             <1>       xchg   ecx, ebx
10464 00010325 F3A4             <1>       rep    movsb
10465                           <1>
10466 00010327 88D9             <1>       mov    cl, bl
10467                           <1>
10468                           <1>       ; 23/01/2021
10469 00010329 EB44             <1>       jmp    short sysvideo_28_10
10470                           <1>
10471                           <1> sysvideo_28_4:
10472 0001032B 53               <1>       push   ebx
10473                           <1>       ; 24/01/2021
10474 0001032C 31DB             <1>       xor    ebx, ebx ; 0 ; use kernel's buffer
10475 0001032E 881D[85120300]   <1>       mov    [srvso], bl ; 0 ; invalidate
10476 00010334 891D[86120300]   <1>       mov    [VideoStateID], ebx ; 0 ; invalidate
10477 0001033A 0FB6CA           <1>       movzx  ecx, dl ; options
10478 0001033D B201             <1>       mov    dl, 1 ; save state
10479 0001033F E890000000       <1>       call   sysvideo_28_11 ; 23/01/2021
10480                           <1>       ; Note: VBE3 BIOS data save option will be
10481                           <1>       ;       disabled.. ; 24/01/2021
10482 00010344 89CA             <1>       mov    edx, ecx ; state (save) options
10483 00010346 5B               <1>       pop    ebx
10484                           <1>
10485 00010347 6683F84F         <1>       cmp    ax, 4Fh ; successful ?
10486 0001034B 7536             <1>       jne    short sysvideo_28_3 ; no !
10487                           <1>
10488 0001034D F6C340           <1>       test   bl, 64 ; verify without transfer
10489 00010350 7536             <1>       jnz    short sysvideo_28_6 ; yes
10490                           <1>
10491                           <1>       ; ecx = cl = options
10492 00010352 E80E39FFFF       <1>       call   vbe_srs_gbs
10493                           <1>       ; ebx = state buffer size (data size)
10494                           <1>
10495 00010357 BE00760900       <1>       mov    esi, VBE3SAVERESTOREBLOCK
10496 0001035C BF00580900       <1>       mov    edi, VBE3VIDEOSTATE
10497 00010361 89D9             <1>       mov    ecx, ebx
10498 00010363 F3A4             <1>       rep    movsb
10499                           <1>
10500 00010365 88D1             <1>       mov    cl, dl
10501 00010367 80C980           <1>       or     cl, 80h ; SVGA (VESA VBE) flag
10502                           <1>       ;mov   [srvso], dl
10503                           <1>
10504 0001036A E908010000       <1>       jmp    sysvideo_28_15
10505                           <1>
10506                           <1>       ; 23/01/2021
10507                           <1> sysvideo_28_10:
10508                           <1>       ; CL = VESA VBE3 Save/Restore options
10509                           <1>
10510 0001036F B202             <1>       mov    dl, 2 ; restore state
10511                           <1>
10512 00010371 E85C000000       <1>       call   sysvideo_28_1
10513                           <1>
10514 00010376 6683F84F         <1>       cmp    ax, 4Fh ; successful ?
10515 0001037A 7407             <1>       je     short sysvideo_28_3
10516                           <1>       ;jmp   short sysvideo_28_9
10517                           <1>
10518                           <1> sysvideo_28_9:
10519                           <1>       ; return zero size (error) to user
10520 0001037C 29C0             <1>       sub    eax, eax
10521                           <1> sysvideo_28_5:
10522 0001037E A3[64030300]     <1>       mov    [u.r0], eax
10523                           <1> sysvideo_28_3:
10524 00010383 E964D5FFFF       <1>       jmp    sysret
10525                           <1>
10526                           <1> sysvideo_28_6:
10527                           <1>       ; use timer ticks as VideoStateID
10528 00010388 A1[10820100]     <1>       mov    eax, [TIMER_LH]
10529 0001038D 09C0             <1>       or     eax, eax
10530 0001038F 75ED             <1>       jnz    short sysvideo_28_5
10531 00010391 40               <1>       inc    eax
10532 00010392 EBEA             <1>       jmp    short sysvideo_28_5
10533                           <1>
10534                           <1> sysvideo_28_7:
10535                           <1>       ; save/restore to/from user buffer
10536                           <1>
10537                           <1>       ; 23/01/2021
10538 00010394 89CE             <1>       mov    esi, ecx ; user's vstate buffer
10539 00010396 BF00760900       <1>       mov    edi, VBE3SAVERESTOREBLOCK
10540                           <1>
10541 0001039B 0FB6CA           <1>       movzx  ecx, dl ; VESA VBE func 4F04h options
10542                           <1>
10543                           <1>       ; source or destination is user buffer
10544 0001039E F6C301           <1>       test   bl, 1
10545 000103A1 7444             <1>       jz     short sysvideo_28_12  ; Save
10546                           <1>
10547                           <1>       ; Restore
10548 000103A3 803D[84120300]00 <1>       cmp    byte [srvsf], 0 ; srs permission flag
10549 000103AA 766A             <1>       jna    short sysvideo_28_14 ; not permitted
10550                           <1>
10551 000103AC 88DA             <1>       mov    dl, bl ; 'sysvideo' options
10552                           <1>
10553                           <1>       ; ecx = cl = options
10554 000103AE E8B238FFFF       <1>       call   vbe_srs_gbs
10555                           <1>       ; ebx = state buffer size (data size)
10556                           <1>
10557 000103B3 891D[64030300]   <1>       mov    [u.r0], ebx ; transfer count
10558                           <1>
10559 000103B9 F6C240           <1>       test   dl, 64 ; verify without transfer
```

```
10560 000103BC 7558                <1>       jnz    short sysvideo_28_14 ; yes
10561                              <1>
10562 000103BE 6681FB0008          <1>       cmp    bx, 2048
10563 000103C3 73B7                <1>       jnb    short sysvideo_28_9 ; invalid
10564                              <1>
10565 000103C5 87CB                <1>       xchg   ecx, ebx
10566                              <1>       ; esi = user buffer
10567                              <1>       ; edi = VBE3SAVERESTOREBLOCK
10568                              <1>
10569 000103C7 E8420F0000          <1>       call   transfer_from_user_buffer
10570 000103CC 72AE                <1>       jc     short sysvideo_28_9 ; error
10571                              <1>
10572 000103CE 89D9                <1>       mov    ecx, ebx ; Function 4F04h options
10573 000103D0 EB9D                <1>       jmp    short sysvideo_28_10 ; 23/01/2021
10574                              <1>
10575                              <1> sysvideo_28_1:
10576 000103D2 31DB                <1>       xor    ebx, ebx ; 0 ; use kernel's buffer
10577                              <1> sysvideo_28_11:
10578                              <1>       ; 24/01/2021
10579 000103D4 803D[54090000]03    <1>       cmp    byte [vbe3], 3
10580 000103DB 7405                <1>       je     short sysvideo_28_2
10581                              <1>
10582                              <1>       ; VESA VBE2 (BOCHS/QEMU/VBOX) video bios
10583 000103DD E9EC37FFFF          <1>       jmp    _vbe_biosfn_save_restore_state
10584                              <1> sysvideo_28_2:
10585                              <1>       ;24/01/2021
10586                              <1>       ;mov   eax, 4F04h  ; Save/Restore vstate
10587                              <1>       ; VESA VBE3 video bios
10588 000103E2 E94F16FFFF          <1>       jmp    _vbe3_pmfn_save_restore_state
10589                              <1>
10590                              <1> sysvideo_28_12:
10591                              <1>       ; Save
10592                              <1>       ;mov   edi, VBE3SAVERESTOREBLOCK
10593                              <1>
10594                              <1>       ;movzx ecx, dl ; options
10595 000103E7 56                  <1>       push   esi
10596 000103E8 53                  <1>       push   ebx
10597                              <1>       ; 23/01/2021
10598 000103E9 B201                <1>       mov    dl, 1 ; save state
10599 000103EB E8E2FFFFFF          <1>       call   sysvideo_28_1
10600 000103F0 5A                  <1>       pop    edx ; 'sysvideo' options
10601 000103F1 5F                  <1>       pop    edi ; user's video state buffer
10602                              <1>
10603 000103F2 6683F84F            <1>       cmp    ax, 4Fh ; successful ?
10604 000103F6 751E                <1>       jne    short sysvideo_28_14 ; no !
10605                              <1>
10606                              <1>       ; ecx = cl = options
10607 000103F8 E86838FFFF          <1>       call   vbe_srs_gbs
10608                              <1>       ; ebx = state buffer size (data size)
10609                              <1>
10610 000103FD 89D9                <1>       mov    ecx, ebx ; transfer count
10611                              <1>
10612 000103FF F6C240              <1>       test   dl, 64 ; verify without transfer
10613 00010402 750C                <1>       jnz    short sysvideo_28_13 ; yes
10614                              <1>
10615                              <1>       ;mov   edi, esi
10616 00010404 BE00760900          <1>       mov    esi, VBE3SAVERESTOREBLOCK
10617 00010409 E8B60E0000          <1>       call   transfer_to_user_buffer
10618 0001040E 7206                <1>       jc     short sysvideo_28_14
10619                              <1> sysvideo_28_13:
10620 00010410 890D[64030300]      <1>       mov    [u.r0], ecx
10621                              <1> sysvideo_28_14:
10622 00010416 E9D1D4FFFF          <1>       jmp    sysret
10623                              <1>
10624                              <1> sysvideo_29:
10625                              <1>       ; 18/01/2021
10626                              <1>       ; BH = 13
10627                              <1>       ; Save/Restore std VGA video state
10628                              <1>
10629                              <1>       ; bl = 0..3
10630                              <1>       ;      save to or restore from
10631                              <1>       ;      system buffer, VBE3VIDEOSTATE
10632                              <1>       ;      ECX = VideoStateID for restoring
10633                              <1>       ; bl = 4..7
10634                              <1>       ;      save to or restore from
10635                              <1>       ;      user buffer pointed by ECX
10636                              <1>
10637 0001041B 80FB03              <1>       cmp    bl, 3
10638 0001041E 7776                <1>       ja     short sysvideo_29_6
10639                              <1>
10640                              <1>       ; source or destination is kernel/system buffer
10641                              <1>
10642 00010420 803D[84120300]00    <1>       cmp    byte [srvsf], 0 ; srs permission flag
10643 00010427 7668                <1>       jna    short sysvideo_29_5 ; not permitted
10644                              <1>
10645 00010429 F6C301              <1>       test   bl, 1
10646 0001042C 7437                <1>       jz     short sysvideo_29_2  ; Save
10647                              <1>
10648                              <1>       ; Restore
10649 0001042E 3B0D[86120300]      <1>       cmp    ecx, [VideoStateID]
10650 00010434 755B                <1>       jne    short sysvideo_29_5 ; not correct ID !
10651 00010436 80FB01              <1>       cmp    bl, 1
10652 00010439 7709                <1>       ja     short sysvideo_29_0
10653                              <1>       ; bl = 1
10654 0001043B BB6E000000          <1>       mov    ebx, 110
10655 00010440 B103                <1>       mov    cl, 3 ; ctrl, vbios data
10656 00010442 EB07                <1>       jmp    short sysvideo_29_1
10657                              <1> sysvideo_29_0:
10658                              <1>       ; bl  = 3
10659 00010444 BB72030000          <1>       mov    ebx, 882
10660 00010449 B107                <1>       mov    cl, 7 ; ctrl, vbios data, dac
10661                              <1> sysvideo_29_1:
10662 0001044B 3A0D[85120300]      <1>       cmp    cl, [srvso]
10663 00010451 753E                <1>       jne    short sysvideo_29_5 ; not correct !
10664                              <1>
```

```
10665 00010453 BE00580900          <1>       mov   esi, VBE3VIDEOSTATE ; 22/01/2021
10666 00010458 E8A239FFFF          <1>       call  biosfn_restore_video_state
10667 0001045D 891D[64030300]      <1>       mov   [u.r0], ebx ; video state size (bytes)
10668                              <1>       ;jmp  sysret
10669 00010463 EB2C                <1>       jmp   short sysvideo_29_5
10670                              <1> sysvideo_29_2:
10671                              <1>       ;mov  esi, ecx
10672 00010465 BF00580900          <1>       mov   edi, VBE3VIDEOSTATE
10673                              <1>
10674 0001046A B107                <1>       mov   cl, 7 ; ctrl, vbios data, dac
10675 0001046C 08DB                <1>       or    bl, bl
10676 0001046E 7502                <1>       jnz   short sysvideo_29_3 ; bl = 2
10677                              <1>       ; bl = 0
10678 00010470 B103                <1>       mov   cl, 3 ; ctrl, vbios data
10679                              <1> sysvideo_29_3:
10680 00010472 E81A38FFFF          <1>       call  biosfn_save_video_state
10681                              <1> sysvideo_28_15:
10682                              <1>       ; use timer ticks as VideoStateID
10683 00010477 A1[10820100]        <1>       mov   eax, [TIMER_LH]
10684 0001047C 21C0                <1>       and   eax, eax
10685 0001047E 7501                <1>       jnz   short sysvideo_29_4
10686 00010480 40                  <1>       inc   eax
10687                              <1> sysvideo_29_4:
10688 00010481 880D[85120300]      <1>       mov   [srvso], cl
10689 00010487 A3[86120300]        <1>       mov   [VideoStateID], eax
10690 0001048C A3[64030300]        <1>       mov   [u.r0], eax
10691                              <1> sysvideo_29_5:
10692 00010491 E956D4FFFF          <1>       jmp   sysret
10693                              <1>
10694                              <1> sysvideo_29_6:
10695 00010496 80FB07              <1>       cmp   bl, 7
10696 00010499 77F6                <1>       ja    short sysvideo_29_5 ; invalid sub function
10697                              <1>
10698 0001049B 89CE                <1>       mov   esi, ecx
10699 0001049D BF00760900          <1>       mov   edi, VBE3SAVERESTOREBLOCK
10700                              <1>
10701                              <1>       ; source or destination is user buffer
10702 000104A2 F6C301              <1>       test  bl, 1
10703 000104A5 7434                <1>       jz    short sysvideo_29_9  ; Save
10704                              <1>
10705                              <1>       ; Restore
10706 000104A7 803D[84120300]00    <1>       cmp   byte [srvsf], 0 ; srs permission flag
10707 000104AE 76E1                <1>       jna   short sysvideo_29_5 ; not permitted
10708                              <1>
10709                              <1>       ;mov  esi, ecx
10710                              <1>       ;mov  edi, VBE3SAVERESTOREBLOCK
10711                              <1>
10712 000104B0 80FB07              <1>       cmp   bl, 7
10713 000104B3 7409                <1>       je    short sysvideo_29_7
10714                              <1>       ; bl = 5
10715 000104B5 B303                <1>       mov   bl, 3
10716 000104B7 B96E000000          <1>       mov   ecx, 110
10717 000104BC EB05                <1>       jmp   short sysvideo_29_8
10718                              <1> sysvideo_29_7:
10719                              <1>       ; bl = 7
10720 000104BE B972030000          <1>       mov   ecx, 882
10721                              <1> sysvideo_29_8:
10722 000104C3 E8460E0000          <1>       call  transfer_from_user_buffer
10723 000104C8 72C7                <1>       jc    short sysvideo_29_5
10724 000104CA 890D[64030300]      <1>       mov   [u.r0], ecx
10725 000104D0 88D9                <1>       mov   cl, bl ; mov cl,7 (mov cl,3)
10726 000104D2 89FE                <1>       mov   esi, edi ; VBE3SAVERESTOREBLOCK
10727                              <1>       ; cl = 3 or 7
10728 000104D4 E82639FFFF          <1>       call  biosfn_restore_video_state
10729 000104D9 EBB6                <1>       jmp   sysvideo_29_5
10730                              <1>       ;jmp  sysret
10731                              <1> sysvideo_29_9:
10732                              <1>       ; Save
10733                              <1>       ;mov  edi, VBE3SAVERESTOREBLOCK
10734                              <1>
10735 000104DB 80FB06              <1>       cmp   bl, 6
10736 000104DE 7409                <1>       je    short sysvideo_29_10
10737                              <1>       ; bl = 4
10738 000104E0 BB6E000000          <1>       mov   ebx, 110
10739 000104E5 B103                <1>       mov   cl, 3 ; ctrl, vbios data
10740 000104E7 EB07                <1>       jmp   short sysvideo_29_11
10741                              <1> sysvideo_29_10:
10742                              <1>       ; bl = 6
10743 000104E9 BB72030000          <1>       mov   ebx, 882
10744 000104EE B107                <1>       mov   cl, 7 ; ctrl, vbios data, dac
10745                              <1> sysvideo_29_11:
10746 000104F0 E89C37FFFF          <1>       call  biosfn_save_video_state
10747                              <1>
10748 000104F5 89D9                <1>       mov   ecx, ebx ; transfer count
10749 000104F7 89F7                <1>       mov   edi, esi
10750 000104F9 BE00760900          <1>       mov   esi, VBE3SAVERESTOREBLOCK
10751                              <1>
10752                              <1>       ;call  transfer_to_user_buffer
10753                              <1>       ;jc    short sysvideo_29_5
10754                              <1>       ;mov   [u.r0], ecx ; transfer count
10755                              <1>       ;;jmp  sysret
10756                              <1>       ;jmp   short sysvideo_29_5
10757                              <1>
10758 000104FE EB1A                <1>       jmp   short sysvideo_29_12
10759                              <1>
10760                              <1> sysvideo_30:
10761 00010500 80FF0F              <1>       cmp   bh, 15
10762 00010503 7722                <1>       ja    short sysvideo_31 ; invalid function
10763                              <1>
10764                              <1>       ; BH = 15
10765                              <1>       ; Copy VESA EDID to user's buffer
10766                              <1>
10767 00010505 803D[25430000]4F    <1>       cmp   byte [edid], 4Fh
10768 0001050C 7519                <1>       jne   short sysvideo_31 ; not ready !
10769                              <1>
```

```
10770                                  <1>       ;and   ecx, ecx
10771                                  <1>       ;jz    short sysvideo_31
10772                                  <1>
10773                                  <1>       ; ecx = user's buffer address
10774 0001050E 89CF                    <1>       mov    edi, ecx
10775 00010510 BE[9C110300]            <1>       mov    esi, edid_info
10776 00010515 B980000000              <1>       mov    ecx, 128 ; 128 bytes
10777                                  <1> sysvideo_29_12:
10778 0001051A E8A50D0000              <1>       call   transfer_to_user_buffer
10779 0001051F 7206                    <1>       jc     short sysvideo_31
10780                                  <1>
10781 00010521 890D[64030300]          <1>       mov    [u.r0], ecx ; EDID size, 128 bytes
10782                                  <1> sysvideo_31:
10783 00010527 E9C0D3FFFF              <1>       jmp    sysret
10784                                  <1>
10785                                  <1> mkdir:
10786                                  <1>       ; 04/12/2015 (14 byte directory names)
10787                                  <1>       ; 12/10/2015
10788                                  <1>       ; 17/06/2015 (Retro UNIX 386 v1 - Beginning)
10789                                  <1>       ; 29/04/2013 - 01/08/2013 (Retro UNIX 8086 v1)
10790                                  <1>       ;
10791                                  <1>       ; 'mkdir' makes a directory entry from the name pointed to
10792                                  <1>       ; by u.namep into the current directory.
10793                                  <1>       ;
10794                                  <1>       ; INPUTS ->
10795                                  <1>       ;    u.namep - points to a file name
10796                                  <1>       ;                 that is about to be a directory entry.
10797                                  <1>       ;    ii - current directory's i-number.
10798                                  <1>       ; OUTPUTS ->
10799                                  <1>       ;    u.dirbuf+2 - u.dirbuf+10 - contains file name.
10800                                  <1>       ;    u.off - points to entry to be filled
10801                                  <1>       ;           in the current directory
10802                                  <1>       ;    u.base - points to start of u.dirbuf.
10803                                  <1>       ;    r1 - contains i-number of current directory
10804                                  <1>       ;
10805                                  <1>       ; ((AX = R1)) output
10806                                  <1>       ;
10807                                  <1>       ;    (Retro UNIX Prototype : 11/11/2012, UNIXCOPY.ASM)
10808                                  <1>         ;    ((Modified registers: eAX, eDX, eBX, eCX, eSI, eDI, eBP))
10809                                  <1>       ;
10810                                  <1>
10811                                  <1>       ; 17/06/2015 - 32 bit modifications (Retro UNIX 386 v1)
10812 0001052C 31C0                    <1>       xor    eax, eax
10813 0001052E BF[9A030300]            <1>       mov     edi, u.dirbuf+2
10814 00010533 89FE                    <1>       mov    esi, edi
10815 00010535 AB                      <1>       stosd
10816 00010536 AB                      <1>       stosd
10817                                  <1>       ; 04/12/2015 (14 byte directory names)
10818 00010537 AB                      <1>       stosd
10819 00010538 66AB                    <1>       stosw
10820                                  <1>             ; jsr r0,copyz; u.dirbuf+2; u.dirbuf+10. / clear this
10821 0001053A 89F7                    <1>       mov    edi, esi ; offset to u.dirbuf
10822                                  <1>       ; 12/10/2015 ([u.namep] -> ebp)
10823                                  <1>       ;mov   ebp, [u.namep]
10824 0001053C E80D030000              <1>       call   trans_addr_nmbp ; convert virtual address to physical
10825                                  <1>             ; esi = physical address (page start + offset)
10826                                  <1>             ; ecx = byte count in the page (1 - 4096)
10827                                  <1>       ; edi = offset to u.dirbuf (edi is not modified in trans_addr_nm)
10828                                  <1>             ; mov u.namep,r2 / r2 points to name of directory entry
10829                                  <1>             ; mov $u.dirbuf+2,r3 / r3 points to u.dirbuf+2
10830                                  <1> mkdir_1: ; 1:
10831 00010541 45                      <1>       inc    ebp ; 12/10/2015
10832                                  <1>       ;
10833                                  <1>       ; / put characters in the directory name in u.dirbuf+2 - u.dirbuf+10
10834                                  <1>        ; 01/08/2013
10835 00010542 AC                      <1>       lodsb
10836                                  <1>             ; movb (r2)+,r1 / move character in name to r1
10837 00010543 20C0                    <1>       and    al, al
10838 00010545 7427                    <1>       jz     short mkdir_3
10839                                  <1>             ; beq 1f / if null, done
10840 00010547 3C2F                    <1>       cmp    al, '/'
10841                                  <1>             ; cmp r1,$'/ / is it a "/"?
10842 00010549 7414                    <1>       je     short mkdir_err
10843                                  <1>       ;je    error
10844                                  <1>             ; beq error9 / yes, error
10845                                  <1>       ; 12/10/2015
10846 0001054B 6649                    <1>       dec    cx
10847 0001054D 7505                    <1>       jnz    short mkdir_2
10848                                  <1>       ; 12/10/2015 ([u.namep] -> ebp)
10849 0001054F E800030000              <1>       call   trans_addr_nm ; convert virtual address to physical
10850                                  <1>             ; esi = physical address (page start + offset)
10851                                  <1>             ; ecx = byte count in the page
10852                                  <1>       ; edi = offset to u.dirbuf (edi is not modified in trans_addr_nm)
10853                                  <1> mkdir_2:
10854 00010554 81FF[A8030300]          <1>       cmp     edi, u.dirbuf+16 ; ; 04/12/2015 (10 -> 16)
10855                                  <1>             ; cmp r3,$u.dirbuf+10. / have we reached the last slot for
10856                                  <1>                                    ; / a char?
10857 0001055A 74E5                    <1>       je     short mkdir_1
10858                                  <1>             ; beq 1b / yes, go back
10859 0001055C AA                      <1>       stosb
10860                                  <1>             ; movb r1,(r3)+ / no, put the char in the u.dirbuf
10861 0001055D EBE2                    <1>       jmp    short mkdir_1
10862                                  <1>             ; br 1b / get next char
10863                                  <1> mkdir_err:
10864                                  <1>       ; 17/06/2015
10865 0001055F C705[C8030300]1300-     <1>       mov    dword [u.error], ERR_NOT_DIR ; 'not a valid directory !'
10865 00010567 0000                    <1>
10866 00010569 E95ED3FFFF              <1>       jmp    error
10867                                  <1>
10868                                  <1> mkdir_3: ; 1:
10869 0001056E A1[78030300]            <1>       mov    eax, [u.dirp]
10870 00010573 A3[80030300]            <1>       mov    [u.off], eax
10871                                  <1>             ; mov u.dirp,u.off / pointer to empty current directory
10872                                  <1>                                ; / slot to u.off
10873                                  <1> wdir: ; 29/04/2013
```

```
10874 00010578 C705[84030300]-     <1>         mov     dword [u.base], u.dirbuf
10874 0001057E [98030300]          <1>
10875                              <1>             ; mov $u.dirbuf,u.base / u.base points to created file name
10876 00010582 C705[88030300]1000- <1>         mov     dword [u.count], 16 ; 04/12/2015 (10 -> 16)
10876 0001058A 0000                <1>
10877                              <1>             ; mov $10.,u.count / u.count = 10
10878 0001058C 66A1[51040300]      <1>       mov    ax, [ii]
10879                              <1>             ; mov ii,r1 / r1 has i-number of current directory
10880 00010592 B201                <1>       mov    dl, 1 ; owner flag mask ; RETRO UNIX 8086 v1 modification !
10881 00010594 E8741D0000          <1>       call   access
10882                              <1>             ; jsr r0,access; 1 / get i-node and set its file up
10883                              <1>                               ; / for writing
10884                              <1>       ; AX = i-number of current directory
10885                              <1>       ; 01/08/2013
10886 00010599 FE05[C6030300]      <1>       inc     byte [u.kcall] ; the caller is 'mkdir' sign
10887 0001059F E8820F0000          <1>       call   writei
10888                              <1>             ; jsr r0,writei / write into directory
10889 000105A4 C3                  <1>       retn
10890                              <1>             ; rts r0
10891                              <1>
10892                              <1> sysexec:
10893                              <1>       ; 18/11/2017
10894                              <1>       ; 14/11/2017
10895                              <1>       ; 13/11/2017
10896                              <1>       ; 24/10/2016, 04/01/2017
10897                              <1>       ; 24/04/2016 - TRDOS 386 (TRDOS v2.0)
10898                              <1>       ; 23/06/2015 - 23/10/2015 (Retro UNIX 386 v1)
10899                              <1>       ; 03/06/2013 - 06/12/2013 (Retro UNIX 8086 v1)
10900                              <1>       ;
10901                              <1>       ; 'sysexec' initiates execution of a file whose path name if
10902                              <1>       ; pointed to by 'name' in the sysexec call.
10903                              <1>       ; 'sysexec' performs the following operations:
10904                              <1>       ;    1. obtains i-number of file to be executed via 'namei'.
10905                              <1>       ;    2. obtains i-node of file to be exceuted via 'iget'.
10906                              <1>       ;    3. sets trap vectors to system routines.
10907                              <1>       ;    4. loads arguments to be passed to executing file into
10908                              <1>       ;      highest locations of user's core
10909                              <1>       ;    5. puts pointers to arguments in locations immediately
10910                              <1>       ;      following arguments.
10911                              <1>       ;    6.      saves number of arguments in next location.
10912                              <1>       ;    7. intializes user's stack area so that all registers
10913                              <1>       ;      will be zeroed and the PS is cleared and the PC set
10914                              <1>       ;      to core when 'sysret' restores registers
10915                              <1>       ;      and does an rti.
10916                              <1>       ;    8. inializes u.r0 and u.sp
10917                              <1>       ;    9. zeros user's core down to u.r0
10918                              <1>       ;   10.      reads executable file from storage device into core
10919                              <1>       ;      starting at location 'core'.
10920                              <1>       ;   11.      sets u.break to point to end of user's code with
10921                              <1>       ;      data area appended.
10922                              <1>       ;   12.      calls 'sysret' which returns control at location
10923                              <1>       ;      'core' via 'rti' instruction.
10924                              <1>       ;
10925                              <1>       ; Calling sequence:
10926                              <1>       ;      sysexec; namep; argp
10927                              <1>       ; Arguments:
10928                              <1>       ;      namep - points to pathname of file to be executed
10929                              <1>       ;      argp  - address of table of argument pointers
10930                              <1>       ;      argp1... argpn - table of argument pointers
10931                              <1>       ;      argp1:<...0> ... argpn:<...0> - argument strings
10932                              <1>       ; Inputs: (arguments)
10933                              <1>       ; Outputs: -
10934                              <1>       ; ............................................................
10935                              <1>       ;
10936                              <1>       ; Retro UNIX 386 v1 modification:
10937                              <1>       ;      User application runs in it's own virtual space
10938                              <1>       ;      which is izolated from kernel memory (and other
10939                              <1>       ;      memory pages) via 80386   paging in ring 3
10940                              <1>       ;      privilige mode. Virtual start address is always 0.
10941                              <1>       ;      User's core memory starts at linear address 400000h
10942                              <1>       ;      (the end of the 1st 4MB).
10943                              <1>       ;
10944                              <1>       ; Retro UNIX 8086 v1 modification:
10945                              <1>       ;      user/application segment and system/kernel segment
10946                              <1>       ;      are different and sysenter/sysret/sysrele routines
10947                              <1>       ;      are different (user's registers are saved to
10948                              <1>       ;      and then restored from system's stack.)
10949                              <1>       ;
10950                              <1>       ;      NOTE: Retro UNIX 8086 v1 'arg2' routine gets these
10951                              <1>       ;            arguments which were in these registers;
10952                              <1>       ;            but, it returns by putting the 1st argument
10953                              <1>       ;            in 'u.namep' and the 2nd argument
10954                              <1>       ;            on top of stack. (1st argument is offset of the
10955                              <1>       ;            file/path name in the user's program segment.)
10956                              <1>
10957                              <1>       ;call  arg2
10958                              <1>       ; * name - 'u.namep' points to address of file/path name
10959                              <1>       ;          in the user's program segment ('u.segmnt')
10960                              <1>       ;          with offset in BX register (as sysopen argument 1).
10961                              <1>       ; * argp - sysexec argument 2 is in CX register
10962                              <1>       ;          which is on top of stack.
10963                              <1>       ;
10964                              <1>             ; jsr r0,arg2 / arg0 in u.namep,arg1 on top of stack
10965                              <1>
10966                              <1>       ; 23/06/2015 (32 bit modifications)
10967                              <1>
10968                              <1>       ;; 13/11/2017
10969                              <1>       ;;mov  [u.namep], ebx ; argument 1
10970                              <1>         ; 18/10/2015
10971 000105A5 890D[4C040300]      <1>       mov     [argv], ecx  ; * ; argument 2
10972                              <1>
10973                              <1>       ; 13/11/2017
10974 000105AB 89DE                <1>       mov    esi, ebx
10975 000105AD E88E210000          <1>       call   set_working_path_x
10976 000105B2 7319                <1>       jnc    short sysexec_0
```

```
10977                                  <1>
10978                                  <1>       ;; 'bad command or file name'
10979                                  <1>       ;mov   eax, ERR_BAD_CMD_ARG ; 01h ; TRDOS 8086
10980                                  <1>
10981                                  <1>       ; 'file not found !' error
10982 000105B4 B802000000               <1>       mov    eax, ERR_NOT_FOUND ; 02h ; TRDOS 8086
10983                                  <1> sysexec_not_found_err:
10984                                  <1> sysexec_access_error:
10985                                  <1> sysexec_ext_error:
10986 000105B9 A3[64030300]             <1>       mov    [u.r0], eax
10987 000105BE A3[C8030300]             <1>       mov    [u.error], eax
10988 000105C3 E84D220000               <1>       call   reset_working_path
10989 000105C8 E9FFD2FFFF               <1>       jmp    error
10990                                  <1>
10991                                  <1> sysexec_0:
10992                                  <1>       ; 13/11/2017
10993                                  <1>       ;mov   esi, FindFile_Name
10994 000105CD 66B80018                 <1>         mov  ax, 1800h ; Only files
10995 000105D1 E89B8EFFFF               <1>       call   find_first_file
10996 000105D6 72E1                     <1>       jc     short sysexec_not_found_err ; eax = 2
10997                                  <1>
10998                                  <1>       ; check_ file attributes
10999                                  <1>       ; (attribute bits = 00ADVSHR) ; 18h = Directory+Volume
11000                                  <1>       ; BL = Attributes byte
11001                                  <1>
11002 000105D8 F6C306                   <1>         test bl, 6  ; system file or hidden file (S+H)
11003                                  <1>       ;jz    short sysexec_0ext
11004 000105DB 7417                     <1>       jz     short sysexec_1 ; yes
11005                                  <1>
11006                                  <1>       ; 13/11/2017
11007                                  <1>       ; /// TRDOS386 permission check for multiuser mode ///
11008                                  <1>       ; SYSTEM file or HIDDEN file !!
11009                                  <1>       ; (Only super user has permission to run this file.)
11010                                  <1>
11011                                  <1>       ; ([u.uid]=0 for super user or root in multiuser mode)
11012                                  <1>       ; ([u.uid]=0 for any users in singleuser mode)
11013 000105DD 803D[B0030300]00         <1>       cmp    byte [u.uid], 0 ; Super User ([u.uid]=0) ?
11014                                  <1>       ;jna   short sysexec_0ext
11015 000105E4 760E                     <1>       jna    short sysexec_1 ; yes
11016                                  <1>
11017                                  <1>       ; 'permission denied !' error
11018 000105E6 B80B000000               <1>         mov  eax, ERR_FILE_ACCESS  ; 11 = ERR_PERM_DENIED
11019 000105EB EBCC                     <1>         jmp  short sysexec_access_error
11020                                  <1>
11021                                  <1> sysexec_not_exf:
11022                                  <1>       ; 'not executable file !' error
11023 000105ED B816000000               <1>       mov    eax, ERR_NOT_EXECUTABLE
11024 000105F2 EBC5                     <1>       jmp    sysexec_ext_error
11025                                  <1>
11026                                  <1> ;sysexec_0ext:
11027                                  <1> sysexec_1:
11028                                  <1>       ; 18/11/2017
11029 000105F4 BE[388B0100]             <1>       mov    esi, FindFile_Name
11030                                  <1>       ; 13/11/2017
11031                                  <1>       ; check program file name extension
11032                                  <1>       ; ('.PRG' for current TRDOS version)
11033 000105F9 E816A9FFFF               <1>       call   check_prg_filename_ext
11034 000105FE 72ED                     <1>       jc     short sysexec_not_exf
11035                                  <1>
11036                                  <1>       ; 18/11/2017
11037 00010600 3C50                     <1>       cmp    al, 'P'
11038 00010602 75E9                     <1>       jne    short sysexec_not_exf
11039                                  <1>
11040                                  <1>       ; '.PRG' extension is OK.
11041                                  <1>       ; Only '.PRG' files are valid program files
11042                                  <1>       ; for current TRDOS 386 version.
11043                                  <1>
11044 00010604 8B15[648B0100]           <1>       mov    edx, [FindFile_DirEntry+DirEntry_FileSize]
11045 0001060A 66A1[5C8B0100]           <1>       mov    ax, [FindFile_DirEntry+DirEntry_FstClusHI]
11046 00010610 C1E010                   <1>       shl    eax, 16
11047 00010613 66A1[628B0100]           <1>       mov    ax, [FindFile_DirEntry+DirEntry_FstClusLO]
11048                                  <1>       ; EAX = First Cluster number
11049                                  <1>       ; EDX = File Size
11050                                  <1>
11051 00010619 A3[51040300]             <1>       mov    [ii], eax
11052 0001061E 8915[55040300]           <1>       mov    [i.size], edx
11053                                  <1>
11054                                  <1> ;sysexec_1:
11055                                  <1>       ; 13/11/2017 - TRDOS 386 (TRDOS v2.0)
11056                                  <1>       ; 24/06/2015 - 23/10/2015 (Retro UNIX 386 v1)
11057                                  <1>         ; Moving arguments to the end of [u.upage]
11058                                  <1>       ; (by regarding page borders in user's memory space)
11059                                  <1>       ;
11060                                  <1>       ; 10/10/2015
11061                                  <1>       ; 21/07/2015
11062 00010624 89E5                     <1>       mov    ebp, esp ; (**)
11063                                  <1>       ; 18/10/2015
11064 00010626 89EF                     <1>       mov    edi, ebp
11065 00010628 B900010000               <1>       mov    ecx, MAX_ARG_LEN ; 256
11066                                  <1>       ;sub   edi, MAX_ARG_LEN ; 256
11067 0001062D 29CF                     <1>       sub    edi, ecx
11068 0001062F 89FC                     <1>       mov    esp, edi ; *!*
11069 00010631 31C0                     <1>       xor    eax, eax
11070 00010633 A3[8C030300]             <1>       mov    [u.nread], eax ; 0
11071 00010638 66A3[4A040300]           <1>       mov    [argc], ax ; 0 ; 13/11/2017
11072 0001063E 49                       <1>       dec    ecx ; 256 - 1
11073 0001063F 890D[88030300]           <1>       mov    [u.count], ecx ; MAX_ARG_LEN - 1 ; 255
11074                                  <1>       ;mov   dword [u.count], MAX_ARG_LEN - 1 ; 255
11075                                  <1> sysexec_2:
11076 00010645 8B35[4C040300]           <1>       mov    esi, [argv] ; 18/10/2015
11077 0001064B E866000000               <1>       call   get_argp
11078 00010650 B904000000               <1>       mov    ecx, 4 ; mov ecx, 4
11079                                  <1> sysexec_3:
11080 00010655 21C0                     <1>       and    eax, eax
11081 00010657 0F846F050000             <1>         jz     sysexec_6
```

```
11082                                  <1>       ; 18/10/2015
11083 0001065D 010D[4C040300]          <1>       add    [argv], ecx ; 4
11084 00010663 66FF05[4A040300]        <1>       inc    word [argc]
11085                                  <1>       ;
11086 0001066A A3[84030300]            <1>       mov    [u.base], eax
11087                                  <1>       ; 23/10/2015
11088 0001066F 66C705[C4030300]00-     <1>       mov    word [u.pcount], 0
11088 00010677 00                      <1>
11089                                  <1> sysexec_4:
11090 00010678 E8E70B0000              <1>       call   cpass ; get a character from user's core memory
11091 0001067D 750E                    <1>         jnz  short sysexec_5
11092                                  <1>              ; (max. 255 chars + null)
11093                                  <1>       ; 18/10/2015
11094 0001067F 28C0                    <1>       sub    al, al
11095 00010681 AA                      <1>       stosb
11096 00010682 FF05[8C030300]          <1>       inc    dword [u.nread]
11097 00010688 E93F050000              <1>       jmp    sysexec_6 ; 24/04/2016
11098                                  <1> sysexec_5:
11099 0001068D AA                      <1>       stosb
11100 0001068E 20C0                    <1>       and    al, al
11101 00010690 75E6                    <1>       jnz    short sysexec_4
11102 00010692 B904000000              <1>       mov    ecx, 4
11103 00010697 390D[48040300]          <1>       cmp    [ncount], ecx ; 4
11104 0001069D 72A6                    <1>       jb     short sysexec_2
11105 0001069F 8B35[44040300]          <1>       mov    esi, [nbase]
11106 000106A5 010D[44040300]          <1>       add    [nbase], ecx ; 4
11107 000106AB 66290D[48040300]        <1>       sub    [ncount], cx
11108 000106B2 8B06                    <1>       mov    eax, [esi]
11109 000106B4 EB9F                    <1>       jmp    short sysexec_3
11110                                  <1>
11111                                  <1> get_argp:
11112                                  <1>       ; 14/11/2017 - TRDOS 386 (TRDOS v2.0)
11113                                  <1>       ; 18/10/2015 (nbase, ncount)
11114                                  <1>       ; 21/07/2015
11115                                  <1>       ; 24/06/2015 (Retro UNIX 386 v1)
11116                                  <1>       ; Get (virtual) address of argument from user's core memory
11117                                  <1>       ;
11118                                  <1>       ; INPUT:
11119                                  <1>       ;      esi = virtual address of argument pointer
11120                                  <1>       ; OUTPUT:
11121                                  <1>       ;      eax = virtual address of argument
11122                                  <1>       ;
11123                                  <1>       ; Modified registers: EAX, EBX, ECX, EDX, ESI
11124                                  <1>       ;
11125 000106B6 833D[BC030300]00        <1>       cmp     dword [u.ppgdir], 0 ; /etc/init ?
11126                                  <1>                                   ; (the caller is kernel)
11127 000106BD 7667                    <1>         jna     short get_argpk
11128                                  <1>       ;
11129 000106BF 89F3                    <1>       mov    ebx, esi
11130 000106C1 E8185CFFFF              <1>       call   get_physical_addr ; get physical address
11131 000106C6 0F8289000000            <1>         jc     get_argp_err
11132 000106CC A3[44040300]            <1>       mov    [nbase], eax ; physical address
11133 000106D1 66890D[48040300]        <1>       mov    [ncount], cx ; remain byte count in page (1-4096)
11134 000106D8 B804000000              <1>       mov    eax, 4 ; 21/07/2015
11135 000106DD 6639C1                  <1>       cmp    cx, ax ; 4
11136 000106E0 735D                    <1>       jnb    short get_argp2
11137 000106E2 89F3                    <1>       mov    ebx, esi
11138 000106E4 01CB                    <1>       add    ebx, ecx
11139 000106E6 E8F35BFFFF              <1>       call   get_physical_addr ; get physical address
11140 000106EB 7268                    <1>       jc     short get_argp_err
11141                                  <1>       ;push  esi
11142 000106ED 89C6                    <1>       mov    esi, eax
11143 000106EF 66870D[48040300]        <1>       xchg   cx, [ncount]
11144 000106F6 8735[44040300]          <1>       xchg   esi, [nbase]
11145 000106FC B504                    <1>       mov    ch, 4
11146 000106FE 28CD                    <1>       sub    ch, cl
11147                                  <1> get_argp0:
11148 00010700 AC                      <1>       lodsb
11149 00010701 6650                    <1>       push   ax
11150 00010703 FEC9                    <1>       dec    cl
11151 00010705 75F9                    <1>         jnz     short get_argp0
11152 00010707 8B35[44040300]          <1>       mov    esi, [nbase]
11153                                  <1>       ; 21/07/2015
11154 0001070D 0FB6C5                  <1>       movzx  eax, ch
11155 00010710 0105[44040300]          <1>       add    [nbase], eax
11156 00010716 662905[48040300]        <1>       sub    [ncount], ax
11157                                  <1> get_argp1:
11158 0001071D AC                      <1>       lodsb
11159 0001071E FECD                    <1>       dec    ch
11160 00010720 7447                    <1>         jz      short get_argp3
11161 00010722 6650                    <1>         push ax
11162 00010724 EBF7                    <1>       jmp     short get_argp1
11163                                  <1> get_argpk:
11164                                  <1>       ; Argument is in kernel's memory space
11165 00010726 66C705[48040300]00-     <1>       mov    word [ncount], PAGE_SIZE ; 4096
11165 0001072E 10                      <1>
11166 0001072F 8935[44040300]          <1>       mov    [nbase], esi
11167 00010735 8305[44040300]04        <1>       add    dword [nbase], 4
11168 0001073C 8B06                    <1>       mov    eax, [esi] ; virtual addr. = physical addr.
11169 0001073E C3                      <1>       retn
11170                                  <1> get_argp2:
11171                                  <1>       ; 21/07/2015
11172                                  <1>       ;mov   eax, 4
11173 0001073F 8B15[44040300]          <1>       mov    edx, [nbase] ; 18/10/2015
11174 00010745 0105[44040300]          <1>       add    [nbase], eax
11175 0001074B 662905[48040300]        <1>       sub    [ncount], ax
11176                                  <1>       ;
11177 00010752 8B02                    <1>       mov    eax, [edx]
11178 00010754 C3                      <1>       retn
11179                                  <1> get_argp_err:
11180 00010755 A3[C8030300]            <1>       mov    [u.error], eax
11181                                  <1>       ; 14/11/2017
11182 0001075A B801000000              <1>       mov    eax, ERR_BAD_CMD_ARG ; 01h ; TRDOS 8086
11183 0001075F A3[64030300]            <1>       mov    [u.r0], eax
11184 00010764 E963D1FFFF              <1>       jmp    error
```

```
11185                                   <1> get_argp3:
11186 00010769 B103                     <1>         mov     cl, 3
11187                                   <1> get_argp4:
11188 0001076B C1E008                   <1>         shl     eax, 8
11189 0001076E 665A                     <1>         pop     dx
11190 00010770 88D0                     <1>         mov     al, dl
11191 00010772 E2F7                     <1>           loop    get_argp4
11192                                   <1>         ;pop   esi
11193 00010774 C3                       <1>         retn
11194                                   <1>
11195                                   <1> sysstat:
11196                                   <1>         ; 13/01/2017 - TRDOS 386 (TRDOS v2.0)
11197                                   <1>         ; temporary !
11198 00010775 B801000000               <1>         mov     eax, ERR_INV_FNUMBER ; 'invalid function number !'
11199 0001077A A3[C8030300]             <1>           mov     [u.error], eax
11200 0001077F A3[64030300]             <1>           mov     [u.r0], eax
11201 00010784 E943D1FFFF               <1>         jmp     error
11202                                   <1>
11203                                   <1> sysfstat:
11204                                   <1>         ; 13/01/2017 - TRDOS 386 (TRDOS v2.0)
11205                                   <1>         ; temporary !
11206 00010789 B801000000               <1>         mov     eax, ERR_INV_FNUMBER ; 'invalid function number !'
11207 0001078E A3[C8030300]             <1>           mov     [u.error], eax
11208 00010793 A3[64030300]             <1>           mov     [u.r0], eax
11209 00010798 E92FD1FFFF               <1>         jmp     error
11210                                   <1>
11211                                   <1> fclose:
11212                                   <1>         ; 06/10/2016 (TRDOS 386 = TRDOS v2.0)
11213                                   <1>         ;
11214                                   <1>         ; 18/06/2015 (Retro UNIX 386 v1 - Beginning)
11215                                   <1>         ;            (32 bit offset pointer modification)
11216                                   <1>         ; 19/04/2013 - 12/01/2014 (Retro UNIX 8086 v1)
11217                                   <1>         ;
11218                                   <1>         ; Given the file descriptor (index to the u.fp list)
11219                                   <1>         ; 'fclose' first gets the i-number of the file via 'getf'.
11220                                   <1>         ; If i-node is active (i-number > 0) the entry in
11221                                   <1>         ; u.fp list is cleared. If all the processes that opened
11222                                   <1>         ; that file close it, then fsp etry is freed and the file
11223                                   <1>         ; is closed. If not a return is taken.
11224                                   <1>         ; If the file has been deleted while open, 'anyi' is called
11225                                   <1>         ; to see anyone else has it open, i.e., see if it is appears
11226                                   <1>         ; in another entry in the fsp table. Upon return from 'anyi'
11227                                   <1>         ; a check is made to see if the file is special.
11228                                   <1>         ;
11229                                   <1>         ; INPUTS ->
11230                                   <1>         ;    r1 - contains the file descriptor (value=0,1,2...)
11231                                   <1>         ;    u.fp - list of entries in the fsp table
11232                                   <1>         ;    fsp - table of entries (4 words/entry) of open files.
11233                                   <1>         ; OUTPUTS ->
11234                                   <1>         ;    r1 - contains the same file descriptor
11235                                   <1>         ;    r2 - contains i-number
11236                                   <1>         ;
11237                                   <1>         ; ((AX = R1))
11238                                   <1>         ; ((Modified registers: eDX, eBX, eCX, eSI, eDI, eBP))
11239                                   <1>         ;
11240                                   <1>         ; Retro UNIX 8086 v1 modification : CF = 1
11241                                   <1>         ;              if i-number of the file is 0. (error)
11242                                   <1>         ;
11243                                   <1>         ; TRDOS 386 (06/10/2016)
11244                                   <1>         ;
11245                                   <1>         ; INPUT:
11246                                   <1>         ;     EAX = File Handle (File Descriptor, File Index)
11247                                   <1>         ;
11248                                   <1>         ; OUTPUT:
11249                                   <1>         ;     CF = 1 -> File not open !
11250                                   <1>         ;     CF = 0 -> OK!
11251                                   <1>         ;          EBX = File Number (System)
11252                                   <1>         ;          [cdev] = Logical DOS Drive Number
11253                                   <1>         ;          EAX = File Handle/Number (user)
11254                                   <1>         ;
11255                                   <1>         ; Modified Registers: EBX
11256                                   <1>
11257 0001079D 50                       <1>         push    eax ; File handle
11258                                   <1>
11259 0001079E E846000000               <1>         call    getf
11260 000107A3 0F8245240000             <1>         jc      device_close ; eax = device number
11261                                   <1>
11262 000107A9 80BB[B6910100]01         <1>         cmp     byte [ebx+OF_MODE], 1 ; open mode ; 0 = empty entry
11263 000107B0 722E                     <1>         jb      short fclose_1            ; 1 = read, 2 = write
11264                                   <1>
11265 000107B2 83F801                   <1>         cmp     eax, 1 ; is the first cluster number > 0
11266 000107B5 7229                     <1>         jb      short fclose_1 ; no, this is empty entry
11267                                   <1>
11268                                   <1> fclose_0:
11269 000107B7 FE8B[CA910100]           <1>         dec     byte [ebx+OF_OPENCOUNT] ; decrement the number of processes
11270                                   <1>                                     ; that have opened the file
11271 000107BD 7921                     <1>         jns     short fclose_1 ; jump if not negative (jump if bit 7 is 0)
11272                                   <1>                       ; if all processes haven't closed the file, return
11273                                   <1>         ;
11274                                   <1>         ; eax ; First cluster
11275 000107BF 31C0                     <1>         xor     eax, eax ; 0
11276 000107C1 8883[B6910100]           <1>         mov     [ebx+OF_MODE], al ; 0 = empty entry
11277                                   <1>         ;mov    [ebx+OF_STATUS], al ; 0 = empty entry
11278 000107C7 66C1E302                 <1>         shl     bx, 2
11279 000107CB 8983[84910100]           <1>         mov     [ebx+OF_FCLUSTER], eax ; 0
11280 000107D1 8983[9C920100]           <1>         mov     [ebx+OF_CCLUSTER], eax ; 0
11281                                   <1>         ;mov    [ebx+OF_CCINDEX], eax ; 0
11282 000107D7 A3[74030300]             <1>         mov     [u.fofp], eax ; 0
11283 000107DC 66C1EB02                 <1>         shr     bx, 2
11284                                   <1> fclose_1: ; 1:
11285 000107E0 58                       <1>         pop     eax ; File handle (File Descriptor, File Index)
11286 000107E1 C680[6A030300]00         <1>         mov     byte [eax+u.fp], 0 ; clear that entry in the u.fp list
11287 000107E8 C3                       <1>         retn
11288                                   <1>
11289                                   <1> getf:
```

```
11290                                  <1>       ; 12/10/2016
11291                                  <1>       ; 11/10/2016
11292                                  <1>       ; 08/10/2016
11293                                  <1>       ; 06/10/2016 (TRDOS 386 = TRDOS v2.0)
11294                                  <1>       ; / get the device number and the i-number of an open file
11295                                  <1>       ; 13/05/2015
11296                                  <1>       ; 11/05/2015 (Retro UNIX 386 v1 - Beginning)
11297                                  <1>       ; 19/04/2013 - 18/11/2013 (Retro UNIX 8086 v1)
11298                                  <1>       ;
11299 000107E9 89C3                   <1>       mov    ebx, eax
11300                                  <1> getf1:
11301 000107EB 83FB0A                 <1>       cmp    ebx, 10
11302 000107EE 730A                   <1>         jnb  short getf2
11303 000107F0 8A9B[6A030300]         <1>       mov    bl, [ebx+u.fp]
11304 000107F6 08DB                   <1>       or     bl, bl
11305 000107F8 7503                   <1>       jnz    short getf3
11306                                  <1> getf2:
11307                                  <1>       ; 'File not open !' error (ax=0)
11308 000107FA 29C0                   <1>       sub    eax, eax
11309 000107FC C3                     <1>       retn
11310                                  <1> getf3:
11311 000107FD F6C380                 <1>       test   bl, 80h
11312 00010800 7530                   <1>       jnz    short getf5 ; device
11313 00010802 FECB                   <1>       dec    bl ; 0 based
11314 00010804 8A83[AC910100]         <1>       mov    al, [ebx+OF_DRIVE]
11315 0001080A A2[46030300]           <1>       mov    [cdev], al
11316 0001080F C0E302                 <1>       shl    bl, 2 ; *4 (dword offset)
11317 00010812 8B83[FC910100]         <1>       mov    eax, [ebx+OF_SIZE]
11318 00010818 A3[55040300]           <1>       mov    [i.size], eax ; file size
11319 0001081D 8D83[D4910100]         <1>       lea    eax, [ebx+OF_POINTER] ;12/10/2016
11320 00010823 A3[74030300]           <1>       mov    [u.fofp], eax
11321 00010828 8B83[84910100]         <1>       mov    eax, [ebx+OF_FCLUSTER]
11322 0001082E C0EB02                 <1>       shr    bl, 2 ; /4 (byte offset)
11323                                  <1> getf4:
11324 00010831 C3                     <1>       retn
11325                                  <1> getf5:
11326                                  <1>       ; get device number
11327 00010832 80E37F                 <1>       and    bl, 7Fh ; 1 to 7Fh
11328 00010835 FECB                   <1>       dec    bl ; 0 based (0 to 7Eh)
11329 00010837 8A83[DE8F0100]         <1>       mov    al, [ebx+DEV_DRIVER]
11330 0001083D 8AAB[488F0100]         <1>       mov    ch, [ebx+DEV_ACCESS]
11331 00010843 8A8B[FC8F0100]         <1>       mov    cl, [ebx+DEV_OPENMODE]
11332 00010849 80E5FE                 <1>       and    ch, 0FEh ; reset bit 0 ; dev_close
11333 0001084C F9                     <1>       stc ; cf = 1
11334 0001084D C3                     <1>       retn
11335                                  <1>
11336                                  <1> trans_addr_nmbp:
11337                                  <1>       ; 18/10/2015
11338                                  <1>       ; 12/10/2015
11339 0001084E 8B2D[7C030300]         <1>       mov    ebp, [u.namep]
11340                                  <1> trans_addr_nm:
11341                                  <1>       ; Convert virtual (pathname) address to physical address
11342                                  <1>       ; (Retro UNIX 386 v1 feature only !)
11343                                  <1>       ; 18/10/2015
11344                                  <1>       ; 12/10/2015 (u.pnbase & u.pncount has been removed from code)
11345                                  <1>       ; 02/07/2015
11346                                  <1>       ; 17/06/2015
11347                                  <1>       ; 16/06/2015
11348                                  <1>       ;
11349                                  <1>       ; INPUTS:
11350                                  <1>       ;      ebp = pathname address (virtual) ; [u.namep]
11351                                  <1>       ;      [u.pgdir] = user's page directory
11352                                  <1>       ; OUTPUT:
11353                                  <1>       ;       esi = physical address of the pathname
11354                                  <1>       ;      ecx = remain byte count in the page
11355                                  <1>       ;
11356                                  <1>       ; (Modified registers: EAX, EBX, ECX, EDX, ESI)
11357                                  <1>       ;
11358 00010854 833D[BC030300]00       <1>         cmp     dword [u.ppgdir], 0  ; /etc/init ? (sysexec)
11359 0001085B 7618                   <1>       jna    short trans_addr_nmk ; the caller is os kernel;
11360                                  <1>                                   ; it is already physical address
11361 0001085D 50                     <1>       push   eax
11362 0001085E 89EB                   <1>       mov    ebx, ebp ; [u.namep] ; pathname address (virtual)
11363 00010860 E8795AFFFF             <1>              call   get_physical_addr ; get physical address
11364 00010865 7204                   <1>       jc     short tr_addr_nm_err
11365                                  <1>       ; 18/10/2015
11366                                  <1>       ; eax = physical address
11367                                  <1>       ; cx = remain byte count in page (1-4096)
11368                                  <1>              ; 12/10/2015 (cx = [u.pncount])
11369 00010867 89C6                   <1>       mov    esi, eax ; 12/10/2015 (esi=[u.pnbase])
11370 00010869 58                     <1>       pop    eax
11371 0001086A C3                     <1>       retn
11372                                  <1>
11373                                  <1> tr_addr_nm_err:
11374 0001086B A3[C8030300]           <1>       mov    [u.error], eax
11375                                  <1>       ;pop   eax
11376 00010870 E957D0FFFF             <1>       jmp    error
11377                                  <1>
11378                                  <1> trans_addr_nmk:
11379                                  <1>       ; 12/10/2015
11380                                  <1>       ; 02/07/2015
11381 00010875 8B35[7C030300]         <1>       mov    esi, [u.namep]  ; [u.pnbase]
11382 0001087B 66B90010               <1>       mov    cx, PAGE_SIZE ; 4096 ; [u.pncount]
11383 0001087F C3                     <1>       retn
11384                                  <1>
11385                                  <1> sysbreak:
11386                                  <1>       ; 18/10/2015
11387                                  <1>       ; 07/10/2015
11388                                  <1>       ; 23/06/2015 (Retro UNIX 386 v1 - Beginning)
11389                                  <1>       ; 20/06/2013 - 24/03/2014 (Retro UNIX 8086 v1)
11390                                  <1>       ;
11391                                  <1>       ; 'sysbreak' sets the programs break points.
11392                                  <1>       ; It checks the current break point (u.break) to see if it is
11393                                  <1>       ; between "core" and the stack (sp). If it is, it is made an
11394                                  <1>       ; even address (if it was odd) and the area between u.break
```

```
11395                                  <1>       ; and the stack is cleared. The new breakpoint is then put
11396                                  <1>       ; in u.break and control is passed to 'sysret'.
11397                                  <1>       ;
11398                                  <1>       ; Calling sequence:
11399                                  <1>       ;     sysbreak; addr
11400                                  <1>       ; Arguments: -
11401                                  <1>       ;
11402                                  <1>       ; Inputs: u.break - current breakpoint
11403                                  <1>       ; Outputs: u.break - new breakpoint
11404                                  <1>       ;     area between old u.break and the stack (sp) is cleared.
11405                                  <1>       ; .........................................................
11406                                  <1>       ;
11407                                  <1>       ; Retro UNIX 8086 v1 modification:
11408                                  <1>       ;     The user/application program puts breakpoint address
11409                                  <1>       ;       in BX register as 'sysbreak' system call argument.
11410                                  <1>       ;     (argument transfer method 1)
11411                                  <1>       ;
11412                                  <1>       ;  NOTE: Beginning of core is 0 in Retro UNIX 8086 v1 !
11413                                  <1>       ;     ((!'sysbreak' is not needed in Retro UNIX 8086 v1!))
11414                                  <1>       ;  NOTE:
11415                                  <1>       ;     'sysbreak' clears extended part (beyond of previous
11416                                  <1>       ;     'u.break' address) of user's memory for original unix's
11417                                  <1>       ;     'bss' compatibility with Retro UNIX 8086 v1 (19/11/2013)
11418                                  <1>
11419                                  <1>             ; mov u.break,r1 / move users break point to r1
11420                                  <1>             ; cmp r1,$core / is it the same or lower than core?
11421                                  <1>             ; blos 1f / yes, 1f
11422                                  <1>       ; 23/06/2015
11423 00010880 8B2D[90030300]          <1>       mov   ebp, [u.break] ; virtual address (offset)
11424                                  <1>       ;and   ebp, ebp
11425                                  <1>       ;jz    short sysbreak_3
11426                                  <1>       ; Retro UNIX 386 v1 NOTE: u.break points to virtual address !!!
11427                                  <1>       ; (Even break point address is not needed for Retro UNIX 386 v1)
11428 00010886 8B15[5C030300]          <1>       mov   edx, [u.sp] ; kernel stack at the beginning of sys call
11429 0001088C 83C20C                  <1>       add   edx, 12 ; EIP -4-> CS -4-> EFLAGS -4-> ESP (user)
11430                                  <1>       ; 07/10/2015
11431 0001088F 891D[90030300]          <1>       mov   [u.break], ebx ; virtual address !!!
11432                                  <1>       ;
11433 00010895 3B1A                    <1>       cmp   ebx, [edx] ; compare new break point with
11434                                  <1>                        ; with top of user's stack (virtual!)
11435 00010897 7323                    <1>       jnb   short sysbreak_3
11436                                  <1>             ; cmp r1,sp / is it the same or higher
11437                                  <1>                     ; / than the stack?
11438                                  <1>             ; bhis 1f / yes, 1f
11439 00010899 89DE                    <1>       mov   esi, ebx
11440 0001089B 29EE                    <1>       sub   esi, ebp ; new break point - old break point
11441 0001089D 761D                    <1>       jna   short sysbreak_3
11442                                  <1>       ;push  ebx
11443                                  <1> sysbreak_1:
11444 0001089F 89EB                    <1>       mov   ebx, ebp
11445 000108A1 E8385AFFFF              <1>       call  get_physical_addr ; get physical address
11446 000108A6 72C3                    <1>       jc    tr_addr_nm_err
11447                                  <1>       ; 18/10/2015
11448 000108A8 89C7                    <1>       mov   edi, eax
11449 000108AA 29C0                    <1>       sub   eax, eax ; 0
11450                                  <1>             ; ECX = remain byte count in page (1-4096)
11451 000108AC 39CE                    <1>       cmp   esi, ecx
11452 000108AE 7302                    <1>       jnb   short sysbreak_2
11453 000108B0 89F1                    <1>       mov   ecx, esi
11454                                  <1> sysbreak_2:
11455 000108B2 29CE                    <1>       sub   esi, ecx
11456 000108B4 01CD                    <1>       add   ebp, ecx
11457 000108B6 F3AA                    <1>       rep   stosb
11458 000108B8 09F6                    <1>       or    esi, esi
11459 000108BA 75E3                    <1>       jnz   short sysbreak_1
11460                                  <1>       ;
11461                                  <1>             ; bit $1,r1 / is it an odd address
11462                                  <1>             ; beq 2f / no, its even
11463                                  <1>             ; clrb (r1)+ / yes, make it even
11464                                  <1>       ; 2: / clear area between the break point and the stack
11465                                  <1>             ; cmp r1,sp / is it higher or same than the stack
11466                                  <1>             ; bhis 1f / yes, quit
11467                                  <1>             ; clr (r1)+ / clear word
11468                                  <1>             ; br 2b / go back
11469                                  <1>       ;pop   ebx
11470                                  <1> sysbreak_3: ; 1:
11471                                  <1>       ;mov   [u.break], ebx ; virtual address !!!
11472                                  <1>             ; jsr r0,arg; u.break / put the "address"
11473                                  <1>                   ; / in u.break (set new break point)
11474                                  <1>             ; br sysret4 / br sysret
11475 000108BC E92BD0FFFF              <1>       jmp   sysret
11476                                  <1>
11477                                  <1> sysseek: ; / moves read write pointer in an fsp entry
11478                                  <1>       ; 06/11/2016 - TRDOS 386 (TRDOS v2.0)
11479                                  <1>       ; 22/06/2015 (Retro UNIX 386 v1 - Beginning)
11480                                  <1>       ; 07/07/2013 - 05/08/2013 (Retro UNIX 8086 v1)
11481                                  <1>       ;
11482                                  <1>       ; 'sysseek' changes the r/w pointer of (3rd word of in an
11483                                  <1>       ; fsp entry) of an open file whose file descriptor is in u.r0.
11484                                  <1>       ; The file descriptor refers to a file open for reading or
11485                                  <1>       ; writing. The read (or write) pointer is set as follows:
11486                                  <1>       ;     * if 'ptrname' is 0, the pointer is set to offset.
11487                                  <1>       ;     * if 'ptrname' is 1, the pointer is set to its
11488                                  <1>       ;       current location plus offset.
11489                                  <1>       ;     * if 'ptrname' is 2, the pointer is set to the
11490                                  <1>       ;       size of file plus offset.
11491                                  <1>       ; The error bit (e-bit) is set for an undefined descriptor.
11492                                  <1>       ;
11493                                  <1>       ; Calling sequence:
11494                                  <1>       ;     sysseek; offset; ptrname
11495                                  <1>       ; Arguments:
11496                                  <1>       ;     offset - number of bytes desired to move
11497                                  <1>       ;              the r/w pointer
11498                                  <1>       ;     ptrname - a switch indicated above
11499                                  <1>       ;
```

```
11500                                  <1>       ; Inputs: r0 - file descriptor
11501                                  <1>       ; Outputs: -
11502                                  <1>       ; ............................................................
11503                                  <1>       ;
11504                                  <1>       ; Retro UNIX 8086 v1 modification:
11505                                  <1>       ;       'sysseek' system call has three arguments; so,
11506                                  <1>       ;     * 1st argument, file descriptor is in BX (BL) register
11507                                  <1>       ;     * 2nd argument, offset is in CX register
11508                                  <1>       ;     * 3rd argument, ptrname/switch is in DX (DL) register
11509                                  <1>
11510 000108C1 E821000000             <1>       call   seektell
11511                                  <1>       ; EAX = Current R/W pointer of the file
11512                                  <1>       ; EBX = [u.fofp]
11513                                  <1>       ; [u.base] = offset (ECX input)
11514                                  <1>
11515 000108C6 0305[84030300]         <1>       add    eax, [u.base]
11516 000108CC 8903                   <1>       mov    [ebx], eax
11517 000108CE E919D0FFFF             <1>       jmp    sysret
11518                                  <1>
11519                                  <1> systell: ; / get the r/w pointer
11520                                  <1>       ; 06/11/2016 - TRDOS 386 (TRDOS v2.0) - temporary !-
11521                                  <1>       ; 22/06/2015 (Retro UNIX 386 v1 - Beginning)
11522                                  <1>       ; 07/07/2013 - 05/08/2013 (Retro UNIX 8086 v1)
11523                                  <1>       ;
11524                                  <1>       ; Retro UNIX 8086 v1 modification:
11525                                  <1>       ; ! 'systell' does not work in original UNIX v1,
11526                                  <1>       ;         it returns with error !
11527                                  <1>       ; Inputs: r0 - file descriptor
11528                                  <1>       ; Outputs: r0 - file r/w pointer
11529                                  <1>
11530                                  <1>       ;xor   ecx, ecx ; 0
11531 000108D3 BA01000000             <1>       mov    edx, 1 ; 05/08/2013
11532                                  <1>       ;call  seektell
11533 000108D8 E810000000             <1>       call   seektell0 ; 05/08/2013
11534                                  <1>       ;; 06/11/2016
11535                                  <1>       ;; mov eax, [ebx]
11536 000108DD A3[64030300]           <1>       mov    [u.r0], eax
11537 000108E2 E905D0FFFF             <1>       jmp    sysret
11538                                  <1>
11539                                  <1> ; Original unix v1 'systell' system call:
11540                                  <1>             ; jsr r0,seektell
11541                                  <1>             ; br error4
11542                                  <1>
11543                                  <1> seektell:
11544                                  <1>       ; 06/11/2016 - TRDOS 386 (TRDOS v2.0)
11545                                  <1>       ; 03/01/2016
11546                                  <1>       ; 22/06/2015 (Retro UNIX 386 v1 - Beginning)
11547                                  <1>       ; 07/07/2013 - 05/08/2013 (Retro UNIX 8086 v1)
11548                                  <1>       ;
11549                                  <1>       ; 'seektell' puts the arguments from sysseek and systell
11550                                  <1>       ; call in u.base and u.count. It then gets the i-number of
11551                                  <1>       ; the file from the file descriptor in u.r0 and by calling
11552                                  <1>       ; getf. The i-node is brought into core and then u.count
11553                                  <1>       ; is checked to see it is a 0, 1, or 2.
11554                                  <1>       ; If it is 0 - u.count stays the same
11555                                  <1>       ;          1 - u.count = offset (u.fofp)
11556                                  <1>       ;        2 - u.count = i.size (size of file)
11557                                  <1>       ;
11558                                  <1>       ; !! Retro UNIX 8086 v1 modification:
11559                                  <1>       ;     Argument 1, file descriptor is in BX;
11560                                  <1>       ;     Argument 2, offset is in CX;
11561                                  <1>       ;     Argument 3, ptrname/switch is in DX register.
11562                                  <1>       ;
11563                                  <1>       ; ((Return -> eax = base for offset (position= base+offset))
11564                                  <1>       ;
11565 000108E7 890D[84030300]         <1>       mov    [u.base], ecx ; offset
11566                                  <1> seektell0:
11567 000108ED 8915[88030300]         <1>       mov    [u.count], edx
11568                                  <1>       ; EBX = file descriptor (file number)
11569 000108F3 E8F3FEFFFF             <1>       call   getf1
11570                                  <1>       ; EAX = First cluster of the file
11571                                  <1>       ; EBX = File number (Open file number)
11572                                  <1>       ; [u.fofp] = Pointer to File pointer
11573                                  <1>       ; [i.size] = File size
11574                                  <1>
11575 000108F8 09C0                   <1>       or     eax, eax
11576 000108FA 7514                   <1>       jnz    short seektell1
11577                                  <1>
11578 000108FC B80A000000             <1>       mov    eax, ERR_FILE_NOT_OPEN
11579 00010901 A3[64030300]           <1>       mov    [u.r0], eax
11580 00010906 A3[C8030300]           <1>       mov    dword [u.error], eax ; 'file not open !'
11581 0001090B E9BCCFFFFF             <1>       jmp    error
11582                                  <1>
11583                                  <1> seektell1:
11584 00010910 8B1D[74030300]         <1>         mov     ebx, [u.fofp]
11585 00010916 803D[88030300]01       <1>       cmp    byte [u.count], 1
11586 0001091D 7705                   <1>       ja     short seektell2
11587 0001091F 7409                   <1>       je     short seektell3
11588 00010921 31C0                   <1>       xor    eax, eax
11589 00010923 C3                     <1>       retn
11590                                  <1>
11591                                  <1> seektell2:
11592 00010924 A1[55040300]           <1>         mov        eax, [i.size]
11593 00010929 C3                     <1>       retn
11594                                  <1>
11595                                  <1> seektell3:
11596 0001092A 8B03                   <1>       mov    eax, [ebx]
11597 0001092C C3                     <1>       retn
11598                                  <1>
11599                                  <1> sysintr: ; / set interrupt handling
11600                                  <1>       ; 22/06/2015 (Retro UNIX 386 v1 - Beginning)
11601                                  <1>       ; 07/07/2013 (Retro UNIX 8086 v1)
11602                                  <1>       ;
11603                                  <1>       ; 'sysintr' sets the interrupt handling value. It puts
11604                                  <1>       ; argument of its call in u.intr then branches into 'sysquit'
```

```
11605                                  <1>       ; routine. u.tty is checked if to see if a control tty exists.
11606                                  <1>       ; If one does the interrupt character in the tty buffer is
11607                                  <1>       ; cleared and 'sysret'is called. If one does not exits
11608                                  <1>       ; 'sysret' is just called.
11609                                  <1>       ;
11610                                  <1>       ; Calling sequence:
11611                                  <1>       ;     sysintr; arg
11612                                  <1>       ; Argument:
11613                                  <1>       ;     arg - if 0, interrupts (ASCII DELETE) are ignored.
11614                                  <1>       ;         - if 1, intterupts cause their normal result
11615                                  <1>       ;           i.e force an exit.
11616                                  <1>       ;         - if arg is a location within the program,
11617                                  <1>       ;           control is passed to that location when
11618                                  <1>       ;           an interrupt occurs.
11619                                  <1>       ; Inputs: -
11620                                  <1>       ; Outputs: -
11621                                  <1>       ; ..........................................................
11622                                  <1>       ;
11623                                  <1>       ; Retro UNIX 8086 v1 modification:
11624                                  <1>       ;      'sysintr' system call sets u.intr to value of BX
11625                                  <1>       ;     then branches into sysquit.
11626                                  <1>       ;
11627 0001092D 66891D[AA030300]       <1>       mov    [u.intr], bx
11628                                  <1>              ; jsr r0,arg; u.intr / put the argument in u.intr
11629                                  <1>              ; br 1f / go into quit routine
11630 00010934 E9B3CFFFFF             <1>       jmp    sysret
11631                                  <1>
11632                                  <1> sysquit:
11633                                  <1>       ; 22/06/2015 (Retro UNIX 386 v1 - Beginning)
11634                                  <1>       ; 07/07/2013 (Retro UNIX 8086 v1)
11635                                  <1>       ;
11636                                  <1>       ; 'sysquit' turns off the quit signal. it puts the argument of
11637                                  <1>       ; the call in u.quit. u.tty is checked if to see if a control
11638                                  <1>       ; tty exists. If one does the interrupt character in the tty
11639                                  <1>       ; buffer is cleared and 'sysret'is called. If one does not exits
11640                                  <1>       ; 'sysret' is just called.
11641                                  <1>       ;
11642                                  <1>       ; Calling sequence:
11643                                  <1>       ;     sysquit; arg
11644                                  <1>       ; Argument:
11645                                  <1>       ;     arg - if 0, this call diables quit signals from the
11646                                  <1>       ;           typewriter (ASCII FS)
11647                                  <1>       ;         - if 1, quits are re-enabled and cause execution to
11648                                  <1>       ;           cease and a core image to be produced.
11649                                  <1>       ;            i.e force an exit.
11650                                  <1>       ;         - if arg is an addres in the program,
11651                                  <1>       ;           a quit causes control to sent to that
11652                                  <1>       ;           location.
11653                                  <1>       ; Inputs: -
11654                                  <1>       ; Outputs: -
11655                                  <1>       ; ..........................................................
11656                                  <1>       ;
11657                                  <1>       ; Retro UNIX 8086 v1 modification:
11658                                  <1>       ;      'sysquit' system call sets u.quit to value of BX
11659                                  <1>       ;     then branches into 'sysret'.
11660                                  <1>       ;
11661 00010939 66891D[AC030300]       <1>       mov    [u.quit], bx
11662 00010940 E9A7CFFFFF             <1>       jmp    sysret
11663                                  <1>              ; jsr r0,arg; u.quit / put argument in u.quit
11664                                  <1>       ;1:
11665                                  <1>              ; mov u.ttyp,r1 / move pointer to control tty buffer
11666                                  <1>                              ; / to r1
11667                                  <1>              ; beq sysret4 / return to user
11668                                  <1>              ; clrb 6(r1) / clear the interrupt character
11669                                  <1>                       ; / in the tty buffer
11670                                  <1>              ; br sysret4 / return to user
11671                                  <1>
11672                                  <1> anyi:
11673                                  <1>       ; 06/10/2016 (TRDOS 386 = TRDOS v2.0)
11674                                  <1>       ; Major Modification!
11675                                  <1>       ; TRDOS 386 does not permit to delete a file while it is open
11676                                  <1>       ; The role of 'anyi' procedure has beeen changed to ensure that.
11677                                  <1>       ;
11678                                  <1>       ; 22/06/2015 (Retro UNIX 386 v1 - Beginning)
11679                                  <1>       ; 25/04/2013 (Retro UNIX 8086 v1)
11680                                  <1>       ;
11681                                  <1>       ; 'anyi' is called if a file deleted while open.
11682                                  <1>       ; "anyi" checks to see if someone else has opened this file.
11683                                  <1>       ;
11684                                  <1>       ; INPUTS ->
11685                                  <1>       ;    r1 - contains an i-number
11686                                  <1>       ;    fsp - start of table containing open files
11687                                  <1>       ;
11688                                  <1>       ; OUTPUTS ->
11689                                  <1>       ;    "deleted" flag set in fsp entry of another occurrence of
11690                                  <1>       ;         this file and r2 points 1st word of this fsp entry.
11691                                  <1>       ;    if file not found - bit in i-node map is cleared
11692                                  <1>       ;                        (i-node is freed)
11693                                  <1>       ;                  all blocks related to i-node are freed
11694                                  <1>       ;                all flags in i-node are cleared
11695                                  <1>       ; ((AX = R1)) input
11696                                  <1>       ;
11697                                  <1>       ;    (Retro UNIX Prototype : 02/12/2012, UNIXCOPY.ASM)
11698                                  <1>         ;    ((Modified registers: eDX, eCX, eBX, eSI, eDI, eBP))
11699                                  <1>       ;
11700                                  <1>       ; / r1 contains an i-number
11701                                  <1>
11702                                  <1>       ; TRDOS 386 (06/10/2016)
11703                                  <1>       ;
11704                                  <1>       ; INPUT:
11705                                  <1>       ;     EAX = First Cluster
11706                                  <1>       ;      DL = Logical DOS Drive Number
11707                                  <1>       ;
11708                                  <1>       ; OUTPUT:
11709                                  <1>       ;     CF = 1 -> EBX = File Handle/Number/Index
```

```
11710                                   <1>       ;      CF = 0 -> EBX = 0
11711                                   <1>       ;
11712                                   <1>       ; Modified Registers: EBX
11713                                   <1>
11714 00010945 31DB                     <1>       xor    ebx, ebx
11715                                   <1> anyi_0:
11716 00010947 80BB[B6910100]00         <1>       cmp    byte [ebx+OF_MODE], 0 ; 0 = empty entry
11717 0001094E 770A                     <1>       ja     short anyi_2 ; 1 (r), 2 (w) or 3 (r&w)
11718                                   <1> anyi_1:
11719 00010950 FEC3                     <1>       inc    bl
11720 00010952 80FB0A                   <1>       cmp    bl, OPENFILES ; max. count of open files
11721 00010955 72F0                     <1>       jb     short anyi_0
11722 00010957 31C0                     <1>       xor    eax, eax
11723 00010959 C3                       <1>       retn
11724                                   <1> anyi_2:
11725 0001095A 3A93[AC910100]           <1>       cmp    dl, [ebx+OF_DRIVE]
11726 00010960 75EE                     <1>       jne    short anyi_1
11727 00010962 66C1E302                 <1>       shl    bx, 2 ; *4 (dword offset)
11728 00010966 3B83[84910100]           <1>       cmp    eax, [ebx+OF_FCLUSTER]
11729 0001096C 7406                     <1>       je     short anyi_3
11730 0001096E 66C1EB02                 <1>       shr    bx, 2 ; /4 (byte offset)
11731 00010972 EBDC                     <1>       jmp    short anyi_1
11732                                   <1> anyi_3:
11733 00010974 66C1EB02                 <1>       shr    bx, 2 ; /4 (bytes offset) (index)
11734 00010978 F9                       <1>       stc
11735 00010979 C3                       <1>       retn
11736                                   <1>
11737                                   <1> ; Retro UNIX 386 v1 Kernel (v0.2) - SYS9.INC
11738                                   <1> ; Last Modification: 09/12/2015
11739                                   <1>
11740                                   <1> syssleep:
11741                                   <1>       ; 29/06/2015 - (Retro UNIX 386 v1)
11742                                   <1>       ; 11/06/2014 - (Retro UNIX 8086 v1)
11743                                   <1>       ;
11744                                   <1>       ; Retro UNIX 8086 v1 feature only
11745                                   <1>       ; (INPUT -> none)
11746                                   <1>       ;
11747 0001097A 0FB61D[B3030300]         <1>       movzx  ebx, byte [u.uno] ; process number
11748 00010981 8AA3[7F000300]           <1>       mov    ah, [ebx+p.ttyc-1] ; current/console tty
11749 00010987 E881190000               <1>       call   sleep
11750 0001098C E95BCFFFFF               <1>       jmp    sysret
11751                                   <1>
11752                                   <1> _vp_clr:
11753                                   <1>       ; Reset/Clear Video Page
11754                                   <1>       ;
11755                                   <1>       ; 30/06/2015 - (Retro UNIX 386 v1)
11756                                   <1>       ; 21/05/2013 - 30/10/2013(Retro UNIX 8086 v1) (U0.ASM)
11757                                   <1>       ;
11758                                   <1>       ; Retro UNIX 8086 v1 feature only !
11759                                   <1>       ;
11760                                   <1>       ; INPUTS ->
11761                                   <1>       ;   BH = video page number
11762                                   <1>       ;
11763                                   <1>       ; OUTPUT ->
11764                                   <1>       ;   none
11765                                   <1>       ; ((Modified registers: eAX, BH, eCX, eDX, eSI, eDI))
11766                                   <1>       ;
11767                                   <1>       ; 04/12/2013
11768 00010991 28C0                     <1>       sub    al, al
11769                                   <1>       ; al = 0 (clear video page)
11770                                   <1>       ; bh = video page ; 13/05/2016
11771 00010993 B407                     <1>       mov    ah, 07h
11772                                   <1>       ; ah = 7 (attribute/color)
11773 00010995 6631C9                   <1>       xor    cx, cx ; 0, left upper column (cl) & row (cl)
11774 00010998 66BA4F18                 <1>       mov    dx, 184Fh ; right lower column & row (dl=24, dh=79)
11775 0001099C E89C16FFFF               <1>       call   _scroll_up
11776                                   <1>       ; bh = video page
11777 000109A1 6631D2                   <1>       xor    dx, dx ; 0 (cursor position)
11778 000109A4 E9EE19FFFF               <1>       jmp    _set_cpos
11779                                   <1>
11780                                   <1> sysmsg:
11781                                   <1>       ; 07/12/2020
11782                                   <1>       ; 05/12/2020
11783                                   <1>       ; 13/05/2016
11784                                   <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
11785                                   <1>       ; 01/07/2015 - 11/11/2015 (Retro UNIX 386 v1)
11786                                   <1>       ; Print user-application message on user's console tty
11787                                   <1>       ;
11788                                   <1>       ; Input -> EBX = Message address
11789                                   <1>       ;         ECX = Message length (max. 255)
11790                                   <1>       ;         DL = Color (IBM PC Rombios color attributes)
11791                                   <1>       ;
11792 000109A9 81F9FF000000            <1>       cmp    ecx, MAX_MSG_LEN ; 255
11793 000109AF 0F8737CFFFFF             <1>       ja     sysret ; nothing to do with big message size
11794 000109B5 08C9                     <1>       or     cl, cl
11795 000109B7 0F842FCFFFFF             <1>       jz     sysret
11796 000109BD 20D2                     <1>       and    dl, dl
11797 000109BF 7502                     <1>       jnz    short sysmsg0
11798 000109C1 B207                     <1>       mov    dl, 07h ; default color
11799                                   <1>              ; (black background, light gray character)
11800                                   <1> sysmsg0:
11801 000109C3 891D[84030300]           <1>       mov    [u.base], ebx
11802 000109C9 8815[BF810100]           <1>       mov    [ccolor], dl ; color attributes
11803 000109CF 89E5                     <1>       mov    ebp, esp
11804 000109D1 31DB                     <1>       xor    ebx, ebx ; 0
11805 000109D3 891D[8C030300]           <1>       mov    [u.nread], ebx ; 0
11806                                   <1>       ;
11807 000109D9 381D[C6030300]           <1>       cmp    [u.kcall], bl ; 0
11808 000109DF 776F                     <1>       ja     short sysmsgk ; Temporary (01/07/2015)
11809                                   <1>       ;
11810 000109E1 890D[88030300]           <1>       mov    [u.count], ecx
11811                                   <1>       ;inc   ecx ; + 00h ; ASCIIZ
11812                                   <1>       ;
11813                                   <1>       ; 07/12/2020
11814                                   <1>       ;add   ecx, 3
```

```
11815 000109E7 6683C103            <1>       add    cx, 3
11816 000109EB 80E1FC              <1>       and    cl, ~3  ; not 3
11817                              <1>       ;
11818 000109EE 29CC                <1>       sub    esp, ecx
11819 000109F0 89E7                <1>       mov    edi, esp
11820 000109F2 89E6                <1>       mov    esi, esp
11821 000109F4 66891D[C4030300]    <1>       mov    [u.pcount], bx ; reset page (phy. addr.) counter
11822                              <1>       ; 11/11/2015
11823 000109FB 8A25[94030300]      <1>       mov    ah, [u.ttyp] ; recent open tty
11824                              <1>       ; 0 = none
11825 00010A01 FECC                <1>       dec    ah
11826 00010A03 790C                <1>       jns    short sysmsg1
11827 00010A05 8A1D[B3030300]      <1>       mov    bl, [u.uno] ; process number
11828 00010A0B 8AA3[7F000300]      <1>       mov    ah, [ebx+p.ttyc-1] ; user's (process's) console tty
11829                              <1> sysmsg1:
11830 00010A11 8825[96030300]      <1>       mov    [u.ttyn], ah
11831                              <1> sysmsg2:
11832 00010A17 E848080000          <1>       call   cpass
11833 00010A1C 7416                <1>       jz     short sysmsg5
11834 00010A1E AA                  <1>       stosb
11835 00010A1F 20C0                <1>       and    al, al
11836 00010A21 75F4                <1>       jnz    short sysmsg2
11837                              <1> sysmsg3:
11838 00010A23 80FC07              <1>       cmp    ah, 7 ; tty number
11839 00010A26 7711                <1>       ja     short sysmsg6 ; serial port
11840 00010A28 E83E000000          <1>       call   print_cmsg ; 05/12/2020
11841                              <1> sysmsg4:
11842 00010A2D 89EC                <1>       mov    esp, ebp
11843 00010A2F E9B8CEFFFF          <1>       jmp    sysret
11844                              <1> sysmsg5:
11845 00010A34 C60700              <1>       mov    byte [edi], 0
11846 00010A37 EBEA                <1>       jmp    short sysmsg3
11847                              <1> sysmsg6:
11848 00010A39 8A06                <1>       mov    al, [esi]
11849 00010A3B E8CD180000          <1>       call   sndc
11850 00010A40 72EB                <1>       jc     short sysmsg4
11851 00010A42 803E00              <1>       cmp    byte [esi], 0  ; 0 is stop character
11852 00010A45 76E6                <1>       jna    short sysmsg4
11853 00010A47 46                  <1>       inc    esi
11854 00010A48 8A25[96030300]      <1>       mov    ah, [u.ttyn]
11855 00010A4E EBE9                <1>       jmp    short sysmsg6
11856                              <1>
11857                              <1> sysmsgk: ; Temporary (01/07/2015)
11858                              <1>       ; The message has been sent by Kernel (ASCIIZ string)
11859                              <1>       ; (ECX -character count- will not be considered)
11860 00010A50 8B35[84030300]      <1>       mov    esi, [u.base]
11861 00010A56 8A25[BE810100]      <1>       mov    ah, [ptty] ; present/current screen (video page)
11862 00010A5C 8825[96030300]      <1>       mov    [u.ttyn], ah
11863 00010A62 C605[C6030300]00    <1>       mov    byte [u.kcall], 0
11864 00010A69 EBB8                <1>       jmp    short sysmsg3
11865                              <1>
11866                              <1> print_cmsg:
11867                              <1>       ; 08/12/2020
11868                              <1>       ; 07/12/2020
11869                              <1>       ; 05/12/2020
11870                              <1>       ; 18/11/2017
11871                              <1>       ; 13/05/2016 - TRDOS 386 (TRDOS v2.0)
11872                              <1>       ; 01/07/2015 (Retro UNIX 386 v1)
11873                              <1>       ;
11874                              <1>       ; print message (on user's console tty)
11875                              <1>       ;      with requested color
11876                              <1>       ;
11877                              <1>       ; INPUTS:
11878                              <1>       ;      esi = message address
11879                              <1>       ;      [u.ttyn] = tty number (0 to 7)
11880                              <1>       ;      [ccolor] = color attributes (IBM PC BIOS colors)
11881                              <1>       ;
11882                              <1>       ; Modified registers: eax, ebx, ecx, edx, esi, edi
11883                              <1>       ; (ebp must be preserved)
11884                              <1>
11885                              <1>       ;mov   bh, ah
11886 00010A6B 8A3D[96030300]      <1>       mov    bh, [u.ttyn]
11887 00010A71 8A1D[BF810100]      <1>       mov    bl, [ccolor] ; * ; 05/12/2020
11888                              <1>
11889                              <1>       ; 05/12/2020
11890 00010A77 803D[1C120300]00    <1>       cmp    byte [pmi32], 0 ; is vbios's 32 bit pmi enabled ?
11891 00010A7E 772E                <1>       ja     short pcmsg5 ; yes
11892                              <1> pcmsg1:
11893                              <1>       ; 08/12/2020
11894 00010A80 8A1D[BF810100]      <1>       mov    bl, [ccolor] ; * (video.s 'u11'&'beep' change BL)
11895                              <1>
11896 00010A86 AC                  <1>       lodsb
11897 00010A87 20C0                <1>       and    al, al  ; 0
11898 00010A89 743A                <1>       jz     short pcmsg2
11899                              <1> pcmsg7:
11900 00010A8B 56                  <1>       push   esi
11901                              <1>       ;mov   bl, [ccolor] ; * (video.s 'u11'&'beep' change BL)
11902                              <1>       ; 05/12/2020
11903                              <1>       ;;mov  bh, [u.ttyn]
11904                              <1>       ;call  _write_tty
11905                              <1>       ;pop   esi
11906                              <1>       ;jmp   short pcmsg1
11907                              <1> ;pcmsg2:
11908                              <1>       ;retn
11909                              <1>
11910                              <1>       ; 07/12/2020
11911 00010A8C 803D[BA6F0000]03    <1>       cmp     byte [CRT_MODE], 3
11912 00010A93 7708                <1>       ja     short pcmsg4
11913                              <1> pcmsg3:
11914 00010A95 E87618FFFF          <1>       call   _write_tty_m3
11915 00010A9A 5E                  <1>       pop    esi
11916 00010A9B EBE3                <1>       jmp    short pcmsg1
11917                              <1> pcmsg4:
11918 00010A9D 803D[BA6F0000]07    <1>         cmp     byte [CRT_MODE], 7
11919 00010AA4 76EF                <1>       jna    short pcmsg3
```

```
11920 00010AA6 E87F25FFFF          <1>       call   vga_write_teletype
11921 00010AAB 5E                  <1>       pop    esi
11922 00010AAC EBD2                <1>       jmp    short pcmsg1
11923                              <1> pcmsg5:
11924                              <1>       ; 07/12/2020
11925 00010AAE 803D[BA6F0000]07    <1>         cmp     byte [CRT_MODE], 7
11926 00010AB5 76C9                <1>       jna    short pcmsg1
11927                              <1>
11928                              <1>       ; 05/12/2020
11929                              <1>       ; writing message by using
11930                              <1>       ; VESA VBE3 video bios protected mode interface
11931                              <1>
11932 00010AB7 B40E                <1>       mov    ah, 0Eh
11933                              <1> pcmsg6:
11934 00010AB9 AC                  <1>       lodsb
11935 00010ABA 20C0                <1>       and    al, al  ; 0
11936 00010ABC 7407                <1>       jz     short pcmsg2
11937                              <1>       ; bh = video page
11938                              <1>       ; ah = 0Eh
11939                              <1>       ; al = character
11940                              <1>       ; bl = color
11941 00010ABE E8470FFFFF          <1>       call   int10h_32bit_pmi
11942 00010AC3 EBF4                <1>       jmp    short pcmsg6
11943                              <1> pcmsg2:
11944 00010AC5 C3                  <1>       retn
11945                              <1>
11946                              <1> sysgeterr:
11947                              <1>       ; 09/12/2015
11948                              <1>       ; 21/09/2015 - (Retro UNIX 386 v1 feature only!)
11949                              <1>       ; Get last error number or page fault count
11950                              <1>       ; (for debugging)
11951                              <1>       ;
11952                              <1>       ; Input -> EBX = return type
11953                              <1>       ;         0 = last error code (which is in 'u.error')
11954                              <1>       ;         FFFFFFFFh = page fault count for running process
11955                              <1>       ;         FFFFFFFEh = total page fault count
11956                              <1>       ;         1 .. FFFFFFFDh = undefined
11957                              <1>       ;
11958                              <1>       ; Output -> EAX = last error number or page fault count
11959                              <1>       ;         (depending on EBX input)
11960                              <1>       ;
11961 00010AC6 21DB                <1>       and    ebx, ebx
11962 00010AC8 750B                <1>       jnz    short glerr_2
11963                              <1> glerr_0:
11964 00010ACA A1[C8030300]        <1>       mov    eax, [u.error]
11965                              <1> glerr_1:
11966 00010ACF A3[64030300]        <1>       mov    [u.r0], eax
11967 00010AD4 C3                  <1>       retn
11968                              <1> glerr_2:
11969 00010AD5 43                  <1>       inc    ebx ; FFFFFFFFh -> 0, FFFFFFFEh -> FFFFFFFFh
11970 00010AD6 74FD                <1>       jz     short glerr_2 ; page fault count for process
11971 00010AD8 43                  <1>       inc    ebx ; FFFFFFFFh -> 0
11972 00010AD9 75EF                <1>       jnz    short glerr_0
11973 00010ADB A1[80050300]        <1>       mov    eax, [PF_Count] ; total page fault count
11974 00010AE0 EBED                <1>         jmp     short glerr_1
11975                              <1> glerr_3:
11976 00010AE2 A1[CC030300]        <1>       mov    eax, [u.pfcount]
11977 00010AE7 EBE6                <1>       jmp    short glerr_1
11978                              <1>
11979                              <1> load_and_run_file:
11980                              <1>       ; 18/11/2017
11981                              <1>       ; 22/01/2017
11982                              <1>       ; 04/01/2017, 07/01/2017
11983                              <1>       ; 24/10/2016
11984                              <1>       ; 24/04/2016, 02/05/2016, 03/05/2016, 06/05/2016
11985                              <1>       ; 23/04/2016 (TRDOS 386 = TRDOS v2.0)
11986                              <1>       ; 23/10/2015 (Retro UNIX 386 v1, 'sysexec')
11987                              <1>       ; 23/06/2015 (Retro UNIX 386 v1 - Beginning)
11988                              <1>       ; 03/06/2013 - 06/12/2013 (Retro UNIX 8086 v1)
11989                              <1>       ; EAX = First Cluster number
11990                              <1>       ; EDX = File Size
11991                              <1>       ; ESI = Argument list address
11992                              <1>       ; [argc] = argument count
11993                              <1>       ; [u.nread] = argument list length
11994                              <1>       ; [esp] = return address to the caller (*)
11995                              <1>       ;
11996 00010AE9 8935[4C040300]      <1>       mov    [argv], esi
11997 00010AEF 8915[55040300]      <1>       mov    [i.size], edx
11998 00010AF5 A3[51040300]        <1>       mov    [ii], eax
11999                              <1>
12000                              <1>       ;sti   ; 07/01/2017
12001                              <1>       ;mov   eax, [k_page_dir]
12002                              <1>       ;mov   [u.pgdir], eax
12003 00010AFA 31C0                <1>       xor    eax, eax ; clc ; *** ; 04/01/2017
12004                              <1>       ;mov   [u.r0], eax ; 0 ; 07/01/2017
12005                              <1>
12006                              <1>       ; 06/05/2016
12007                              <1>       ; Set 'sysexit' return order to MainProg
12008                              <1>       ;
12009 00010AFC 58                  <1>       pop    eax ; * 'loc_load_and_run_file_8:' address
12010                              <1>       ;; 22/01/2017
12011                              <1>       ;;cli ; 07/01/2017
12012 00010AFD 8B25[2C810100]      <1>       mov    esp, [tss.esp0]
12013                              <1>       ;
12014                              <1>       ; 'loc_load_run_file_8' address has
12015                              <1>       ; 'jmp loc_file_rw_restore_retn' instruction
12016                              <1>       ; 'loc_file_rw_restore_retn:' will return to
12017                              <1>       ; [mainprog_return_addr]
12018                              <1>       ; just after 'call command_interpreter'
12019                              <1>       ;
12020 00010B03 68[23750000]        <1>       push   _end_of_mainprog ; we must not return to here !
12021 00010B08 FF35[108E0100]      <1>       push   dword [mainprog_return_addr]
12022 00010B0E 89E5                <1>       mov    ebp, esp ; **
12023                              <1>       ;
12024 00010B10 9C                  <1>       pushfd  ; EFLAGS      ; IRETD ; ***
```

```
12025 00010B11 6A08              <1>       push   KCODE ; cs    ; IRETD
12026 00010B13 50                <1>       push   eax ; * (eip) ; IRETD
12027 00010B14 8925[5C030300]    <1>       mov    [u.sp], esp
12028                            <1>       ;mov   byte [u.quant], time_count
12029 00010B1A 1E                <1>       push   ds
12030 00010B1B 06                <1>       push   es
12031 00010B1C 0FA0              <1>       push   fs
12032 00010B1E 0FA8              <1>       push   gs
12033                            <1>       ;mov   eax, [u.r0]
12034 00010B20 29C0              <1>       sub    eax, eax
12035 00010B22 60                <1>       pushad
12036 00010B23 68[ECD80000]      <1>       push   sysret
12037                            <1>       ;push  sysrel1 ; 07/01/2017
12038 00010B28 8925[60030300]    <1>       mov    [u.usp], esp
12039                            <1>       ;
12040 00010B2E E845060000        <1>       call   wswap ; Save MainProg (process 1) 'u' structure
12041                            <1>                    ; and registers for return (from program)
12042 00010B33 89EC              <1>       mov    esp, ebp ; **
12043                            <1>       ;;22/01/2017
12044                            <1>       ;;sti ; 07/01/2017
12045 00010B35 50                <1>       push   eax  ; * 'loc_load_and_run_file_8:' address
12046                            <1>       ;
12047                            <1>       ;;; 02/05/2016
12048                            <1>       ;;; Create a new process (parent: MainProg)
12049 00010B36 31F6              <1>       xor    esi, esi
12050                            <1> cnpm_1: ; search p.stat table for unused process number
12051 00010B38 46                <1>       inc    esi
12052 00010B39 80BE[AF000300]00  <1>       cmp    byte [esi+p.stat-1], 0 ; SFREE
12053                            <1>                             ; is process active, unused, dead
12054 00010B40 760B              <1>       jna    short cnpm_2 ; it's unused so branch
12055 00010B42 6683FE10          <1>       cmp    si, nproc    ; all processes checked
12056 00010B46 72F0              <1>       jb     short cnpm_1    ; no, branch back
12057 00010B48 E95E6AFFFF        <1>       jmp    panic
12058                            <1> cnpm_2:
12059 00010B4D A1[B8030300]      <1>       mov    eax, [u.pgdir] ; page directory of MainProg
12060 00010B52 A3[BC030300]      <1>       mov    [u.ppgdir], eax ; parent's page directory
12061 00010B57 E86D50FFFF        <1>       call   allocate_page
12062 00010B5C 0F82496AFFFF      <1>       jc     panic
12063                            <1>       ; EAX = UPAGE (user structure page) address
12064 00010B62 A3[B4030300]      <1>       mov    [u.upage], eax ; memory page for 'user' struct (child)
12065 00010B67 89F7              <1>       mov    edi, esi
12066 00010B69 66C1E702          <1>       shl    di, 2
12067 00010B6D 8987[BC000300]    <1>       mov    [edi+p.upage-4], eax ; memory page for 'user' struct
12068 00010B73 E8CB50FFFF        <1>       call   clear_page ; 03/05/2016
12069                            <1>       ;movzx eax, byte [p.ttyc] ; console tty (for MainProg)
12070 00010B78 6629C0            <1>       sub    ax, ax ; 0
12071 00010B7B 668986[7F000300]  <1>       mov     [esi+p.ttyc-1], ax ; al - set child's console tty
12072                            <1>                               ; ah - reset child's wait channel
12073 00010B82 89F0              <1>       mov    eax, esi
12074 00010B84 A2[B3030300]      <1>       mov    [u.uno], al ; child process number
12075 00010B89 FE86[AF000300]    <1>         inc     byte [esi+p.stat-1] ; 1, SRUN
12076 00010B8F 66D1E6            <1>       shl    si, 1 ; multiply si by 2 to get index into p.pid table
12077 00010B92 66FF05[4E030300]  <1>       inc    word [mpid] ; increment m.pid; get a new process name
12078 00010B99 66A1[4E030300]    <1>       mov    ax, [mpid]
12079 00010B9F 668986[1E000300]  <1>       mov    [esi+p.pid-2], ax ; put new process name
12080                            <1>                              ; in child process' name slot
12081                            <1>       ;mov   ax, [p.pid]  ; get process name of MainProg
12082 00010BA6 66B80100          <1>       mov    ax, 1
12083 00010BAA 668986[3E000300]  <1>       mov    [esi+p.ppid-2], ax ; put parent process name
12084                            <1>                                ; in parent process slot for child
12085 00010BB1 6648              <1>       dec    ax ; 0
12086 00010BB3 66A3[94030300]    <1>       mov    [u.ttyp], ax ; 0
12087                            <1>       ;;;
12088 00010BB9 A1[51040300]      <1>       mov    eax,  [ii]
12089                            <1>       ; Retro UNIX 386 v1, 'sysexec' (u2.s)
12090 00010BBE E84A170000        <1>       call   iopen
12091                            <1>       ; 06/06/2016
12092 00010BC3 C605[A9030300]01  <1>       mov    byte [u.pri], 1 ; normal priority
12093                            <1>       ;
12094 00010BCA EB16              <1>       jmp    short sysexec_7 ; 02/05/2016
12095                            <1>
12096                            <1> sysexec_6:
12097                            <1>       ; 19/11/2017
12098                            <1>       ; 18/11/2017
12099                            <1>       ; 14/11/2017
12100                            <1>       ; 13/11/2017
12101 00010BCC 8925[4C040300]    <1>       mov    [argv], esp ; *!* ; start address of argument list
12102                            <1>
12103                            <1>       ; 04/01/2017
12104                            <1>       ; 24/10/2016
12105                            <1>       ;;02/05/2016
12106                            <1>       ; 23/04/2016 (TRDOS 386)
12107                            <1>       ; 18/10/2015 ('sysexec_6')
12108                            <1>       ; 23/06/2015
12109 00010BD2 A1[B8030300]      <1>       mov    eax, [u.pgdir] ; physical address of page directory
12110                            <1>       ;cmp   eax, [k_page_dir] ; TRDOS MainProg ?
12111                            <1>       ;je    short sysexec_7
12112                            <1>       ; 19/11/2017
12113 00010BD7 8B1D[BC030300]    <1>       mov    ebx, [u.ppgdir] ; phy addr of the parent's page dir
12114 00010BDD E82051FFFF        <1>       call   deallocate_page_dir
12115                            <1> sysexec_7:
12116 00010BE2 E85050FFFF        <1>       call   make_page_dir
12117 00010BE7 0F82BE69FFFF      <1>       jc     panic  ; allocation error
12118                            <1>                     ; after a deallocation would be nonsence !?
12119                            <1>       ; 24/07/2015
12120                            <1>       ; map kernel pages (1st 4MB) to PDE 0
12121                            <1>       ;     of the user's page directory
12122                            <1>       ;     (It is needed for interrupts!)
12123                            <1>       ; 18/10/2015
12124 00010BED 8B15[90810100]    <1>       mov    edx, [k_page_dir] ; Kernel's page directory
12125 00010BF3 8B02              <1>       mov    eax, [edx] ; physical address of
12126                            <1>                       ; kernel's first page table (1st 4 MB)
12127                            <1>                       ; (PDE 0 of kernel's page directory)
12128 00010BF5 8B15[B8030300]    <1>       mov    edx, [u.pgdir]
12129 00010BFB 8902              <1>       mov    [edx], eax ; PDE 0 (1st 4MB)
```

```
12130                                   <1>       ;
12131                                   <1>       ; 20/07/2015
12132 00010BFD BB00004000               <1>       mov    ebx, CORE ; start address = 0 (virtual) + CORE
12133                                   <1>       ; 18/10/2015
12134 00010C02 BE[3C040300]             <1>       mov    esi, pcore ; physical start address
12135                                   <1> sysexec_8:
12136 00010C07 B907000000               <1>       mov    ecx, PDE_A_USER + PDE_A_WRITE + PDE_A_PRESENT
12137 00010C0C E84450FFFF               <1>       call   make_page_table
12138 00010C11 0F829469FFFF             <1>       jc     panic
12139                                   <1>       ;mov   ecx, PTE_A_USER + PTE_A_WRITE + PTE_A_PRESENT
12140 00010C17 E84750FFFF               <1>       call   make_page ; make new page, clear and set the pte
12141 00010C1C 0F828969FFFF             <1>       jc     panic
12142                                   <1>       ;
12143 00010C22 8906                     <1>       mov    [esi], eax ; 24/06/2015
12144                                   <1>       ; ebx = virtual address (24/07/2015)
12145 00010C24 E8DF55FFFF               <1>       call   add_to_swap_queue
12146                                   <1>       ; 18/10/2015
12147 00010C29 81FE[40040300]           <1>       cmp    esi, ecore ; user's stack (last) page ?
12148 00010C2F 740C                     <1>       je     short sysexec_9 ; yes
12149 00010C31 BE[40040300]             <1>       mov    esi, ecore  ; physical address of the last page
12150                                   <1>       ; 20/07/2015
12151 00010C36 BB00F0FFFF               <1>       mov    ebx, (ECORE - PAGE_SIZE) + CORE
12152                                   <1>       ; ebx = virtual end address + segment base address - 4K
12153 00010C3B EBCA                     <1>         jmp     short sysexec_8
12154                                   <1> sysexec_9:
12155                                   <1>       ; 19/11/2017
12156                                   <1>       ; 24/04/2016 (TRDOS 386 = TRDOS v2.0)
12157                                   <1>       ; 25/06/2015, 26/08/2015, 18/10/2015
12158                                   <1>       ; move arguments from kernel stack to [ecore]
12159                                   <1>       ; (argument list/line will be copied from kernel stack
12160                                   <1>       ; frame to the last (stack) page of user's core memory)
12161                                   <1>       ; 18/10/2015
12162 00010C3D 8B3D[40040300]           <1>       mov    edi, [ecore]
12163 00010C43 81C700100000             <1>       add    edi, PAGE_SIZE
12164                                   <1>       ; 19/11/2017
12165 00010C49 83EF04                   <1>       sub    edi, 4
12166 00010C4C C70700000000             <1>       mov    dword [edi], 0
12167 00010C52 89FB                     <1>       mov    ebx, edi
12168                                   <1>       ;
12169 00010C54 0FB705[4A040300]         <1>       movzx  eax, word [argc]
12170 00010C5B 09C0                     <1>       or     eax, eax
12171 00010C5D 7445                     <1>       jz     short sysexec_13 ; 19/11/2017
12172                                   <1>       ;jnz   short sysexec_10
12173                                   <1>       ;mov   ebx, edi
12174                                   <1>       ;sub   ebx, 4
12175                                   <1>       ;mov   [ebx], eax ; 0
12176                                   <1>       ;jmp   short sysexec_13
12177                                   <1> sysexec_10:
12178 00010C5F 8B0D[8C030300]           <1>       mov    ecx, [u.nread]
12179                                   <1>       ; 13/11/2017
12180                                   <1>       ;mov   esi, TextBuffer ; 'load_and_execute_file'
12181                                   <1>       ;mov   esi, esp     ; 'sysexec'
12182 00010C65 8B35[4C040300]           <1>       mov    esi, [argv] ; 24/04/2016 (TRDOS 386  = TRDOS v2.0)
12183                                   <1>       ;sub   edi, ecx ; page end address - argument list length
12184 00010C6B 29CB                     <1>       sub    ebx, ecx ; 19/11/2017
12185 00010C6D 89C2                     <1>       mov    edx, eax
12186 00010C6F FEC2                     <1>       inc    dl ; argument count + 1 for argc value
12187 00010C71 C0E202                   <1>       shl    dl, 2  ; 4 * (argument count + 1)
12188                                   <1>       ;mov   ebx, edi
12189 00010C74 89DF                     <1>       mov    edi, ebx ; 19//11/2017
12190 00010C76 80E3FC                   <1>       and    bl, 0FCh ; 32 bit (dword) alignment
12191 00010C79 29D3                     <1>       sub    ebx, edx
12192 00010C7B 89FA                     <1>       mov    edx, edi
12193 00010C7D F3A4                     <1>       rep    movsb
12194 00010C7F 89D6                     <1>       mov    esi, edx
12195 00010C81 89DF                     <1>       mov    edi, ebx
12196 00010C83 BA00F0BFFF               <1>       mov    edx, ECORE - PAGE_SIZE ; virtual addr. of the last page
12197 00010C88 2B15[40040300]           <1>       sub    edx, [ecore] ; difference (virtual - physical)
12198 00010C8E AB                       <1>       stosd  ; eax = argument count
12199                                   <1> sysexec_11:
12200 00010C8F 89F0                     <1>       mov    eax, esi
12201 00010C91 01D0                     <1>       add    eax, edx
12202 00010C93 AB                       <1>       stosd  ; eax = virtual address
12203                                   <1>       ;dec   byte [argc]
12204 00010C94 66FF0D[4A040300]         <1>       dec    word [argc] ; 14/11/2017
12205 00010C9B 7407                     <1>       jz     short sysexec_13
12206                                   <1> sysexec_12:
12207 00010C9D AC                       <1>       lodsb
12208 00010C9E 20C0                     <1>       and    al, al
12209 00010CA0 75FB                     <1>       jnz    short sysexec_12
12210 00010CA2 EBEB                     <1>       jmp    short sysexec_11
12211                                   <1> sysexec_13:
12212                                   <1>       ; 24/10/2016
12213                                   <1>       ; 24/04/2016 - TRDOS 386 (TRDOS v2.0)
12214                                   <1>       ; 23/06/2015 - 19/10/2015 (Retro UNIX 386 v1, 'sysexec_13')
12215                                   <1>       ;
12216                                   <1>       ; moving arguments to [ecore] is OK here..
12217                                   <1>       ;
12218                                   <1>       ; ebx = beginning addres of argument list pointers
12219                                   <1>             ;      in user's stack
12220 00010CA4 2B1D[40040300]           <1>       sub    ebx, [ecore]
12221 00010CAA 81C300F0BFFF             <1>       add     ebx, (ECORE - PAGE_SIZE)
12222                                   <1>                   ; end of core - 4096 (last page)
12223                                   <1>                   ; (virtual address)
12224 00010CB0 891D[4C040300]           <1>       mov    [argv], ebx
12225 00010CB6 891D[90030300]           <1>       mov    [u.break], ebx ; available user memory
12226                                   <1>       ;
12227 00010CBC 29C0                     <1>       sub    eax, eax
12228 00010CBE C705[88030300]2000-      <1>       mov    dword [u.count], 32 ; Executable file header size
12228 00010CC6 0000                     <1>
12229 00010CC8 C705[74030300]-          <1>       mov    dword [u.fofp], u.off
12229 00010CCE [80030300]               <1>
12230 00010CD2 A3[80030300]             <1>       mov    [u.off], eax ; 0
12231 00010CD7 A3[84030300]             <1>       mov    [u.base], eax ; 0, start of user's core (virtual)
12232                                   <1>       ; 24/10/2016
```

```
12233 00010CDC A0[56820100]       <1>       mov    al, [Current_Drv]
12234 00010CE1 A2[46030300]       <1>       mov    [cdev], al
12235                             <1>       ;
12236 00010CE6 A1[51040300]       <1>       mov    eax, [ii] ; Fist Cluster of the Program (PRG) file
12237                             <1>       ; EAX = First cluster of the executable file
12238 00010CEB E80A010000         <1>       call   readi
12239                             <1>
12240 00010CF0 8B0D[90030300]     <1>       mov    ecx, [u.break] ; top of user's stack (physical addr.)
12241 00010CF6 890D[88030300]     <1>       mov    [u.count], ecx ; save for overrun check
12242                             <1>       ;
12243 00010CFC 8B0D[8C030300]     <1>       mov    ecx, [u.nread]
12244 00010D02 890D[90030300]     <1>       mov    [u.break], ecx ; virtual address (offset from start)
12245 00010D08 80F920             <1>       cmp    cl, 32
12246 00010D0B 7540               <1>         jne     short sysexec_15
12247                             <1>       ;:
12248                             <1>       ; Retro UNIX 386 v1 (32 bit) executable file header format
12249 00010D0D 8B35[3C040300]     <1>       mov    esi, [pcore] ; start address of user's core memory
12250                             <1>                           ; (phys. start addr. of the exec. file)
12251 00010D13 AD                 <1>       lodsd
12252 00010D14 663DEB1E           <1>       cmp    ax, 1EEBh ; EBH, 1Eh -> jump to +32
12253 00010D18 7533               <1>       jne    short sysexec_15
12254 00010D1A AD                 <1>       lodsd
12255 00010D1B 89C1               <1>       mov    ecx, eax ; text (code) section size
12256 00010D1D AD                 <1>       lodsd
12257 00010D1E 01C1               <1>       add    ecx, eax ; + data section size (initialized data)
12258 00010D20 89CB               <1>       mov    ebx, ecx
12259 00010D22 AD                 <1>       lodsd
12260 00010D23 01C3               <1>       add    ebx, eax ; + bss section size (for overrun checking)
12261 00010D25 3B1D[88030300]     <1>       cmp    ebx, [u.count]
12262 00010D2B 7711               <1>       ja     short sysexec_14  ; program overruns stack !
12263                             <1>       ;
12264                             <1>       ; add bss section size to [u.break]
12265 00010D2D 0105[90030300]     <1>       add    [u.break], eax
12266                             <1>       ;
12267 00010D33 83E920             <1>       sub    ecx, 32  ; header size (already loaded)
12268                             <1>       ;cmp   ecx, [u.count]
12269                             <1>       ;jnb   short sysexec_16
12270 00010D36 890D[88030300]     <1>       mov    [u.count], ecx ; required read count
12271 00010D3C EB29               <1>       jmp    short sysexec_16
12272                             <1> sysexec_14:
12273                             <1>       ; insufficient (out of) memory
12274 00010D3E C705[C8030300]0400- <1>      mov    dword [u.error], ERR_MINOR_IM ; 1
12274 00010D46 0000               <1>
12275 00010D48 E97FCBFFFF         <1>       jmp    error
12276                             <1> sysexec_15:
12277 00010D4D 8B15[55040300]     <1>         mov  edx, [i.size] ; file size
12278 00010D53 29CA               <1>       sub    edx, ecx ; file size - loaded bytes
12279 00010D55 7626               <1>       jna    short sysexec_17 ; no need to next read
12280 00010D57 01D1               <1>       add    ecx, edx ; [i.size]
12281 00010D59 3B0D[88030300]     <1>       cmp    ecx, [u.count] ; overrun check (!)
12282 00010D5F 77DD               <1>       ja     short sysexec_14
12283 00010D61 8915[88030300]     <1>       mov    [u.count], edx
12284                             <1> sysexec_16:
12285 00010D67 A1[51040300]       <1>       mov    eax, [ii] ; first cluster
12286 00010D6C E889000000         <1>       call   readi
12287 00010D71 8B0D[8C030300]     <1>       mov    ecx, [u.nread]
12288 00010D77 010D[90030300]     <1>       add    [u.break], ecx
12289                             <1> sysexec_17:
12290 00010D7D A1[51040300]       <1>       mov    eax, [ii] ; first cluster
12291 00010D82 E886150000         <1>       call   iclose
12292 00010D87 31C0               <1>       xor     eax, eax
12293 00010D89 FEC0               <1>       inc    al
12294 00010D8B 66A3[AA030300]     <1>       mov    [u.intr], ax ; 1 (interrupt/time-out is enabled)
12295 00010D91 66A3[AC030300]     <1>       mov    [u.quit], ax ; 1 ('crtl+brk' signal is enabled)
12296 00010D97 833D[BC030300]00   <1>         cmp  dword [u.ppgdir], 0  ; is the caller MainProg (kernel) ?
12297 00010D9E 770C               <1>       ja     short sysexec_18 ; no, the caller is user process
12298                             <1>       ; If the caller is kernel (MainProg), 'sysexec' will come here
12299 00010DA0 8B15[90810100]     <1>       mov    edx, [k_page_dir] ; kernel's page directory
12300 00010DA6 8915[BC030300]     <1>       mov    [u.ppgdir], edx ; next time 'sysexec' must not come here
12301                             <1> sysexec_18:
12302                             <1>       ; 02/05/2016
12303                             <1>       ; 24/04/2016 (TRDOS 386 = TRDOS v2.0)
12304                             <1>       ; 18/10/2015 (Retro UNIX 386 v1)
12305                             <1>       ; 05/08/2015
12306                             <1>       ; 29/07/2015
12307                             <1>
12308                             <1> ;     ; **** arguments list test start - 19/11/2017
12309                             <1> ;     mov    ebp, [argv]
12310                             <1> ;     sub    ebp, ECORE - 4096
12311                             <1> ;     add    ebp, [ecore]
12312                             <1> ;
12313                             <1> ;     mov    ebx, [ebp]
12314                             <1> ;     mov    [argc], bx
12315                             <1> ;     add    ebp, 4
12316                             <1> ;     mov    byte [ccolor], 1Fh
12317                             <1> ;_zx0:
12318                             <1> ;     cmp    word [argc], 0
12319                             <1> ;     jna    short _zx2
12320                             <1> ;_zx1:
12321                             <1> ;     push   ebp
12322                             <1> ;     mov    esi, [ebp]
12323                             <1> ;
12324                             <1> ;     sub    esi, ECORE - 4096
12325                             <1> ;     add    esi, [ecore]
12326                             <1> ;
12327                             <1> ;     call   print_cmsg
12328                             <1> ;
12329                             <1> ;     dec    word [argc]
12330                             <1> ;     jz     short _zx2
12331                             <1> ;
12332                             <1> ;     mov    al, '.'
12333                             <1> ;     mov    bl, 07h
12334                             <1> ;     mov    bh, [u.ttyn]
12335                             <1> ;     call   _write_tty
12336                             <1> ;
```

```
12337                                  <1> ;       pop     ebp
12338                                  <1> ;       add     ebp, 4
12339                                  <1> ;       jmp     short _zx1
12340                                  <1> ;_zx2:
12341                                  <1> ;       pop     ebp
12342                                  <1> ;       mov     byte [ccolor], 07h
12343                                  <1> ;       mov     eax, 1
12344                                  <1> ;       ; **** arguments list test stop
12345                                  <1> ;       Test result is OK! (there is not a wrong thing) - 19/11/2017
12346                                  <1>
12347 00010DAC 8B2D[4C040300]          <1>         mov     ebp, [argv] ; user's stack pointer must point to argument
12348                                  <1>                             ; list pointers (argument count)
12349 00010DB2 FA                      <1>         cli
12350 00010DB3 8B25[2C810100]          <1>           mov     esp, [tss.esp0] ; ring 0 (kernel) stack pointer
12351                                  <1>         ;mov         esp, [u.sp] ; Restore Kernel stack
12352                                  <1>                             ; for this process
12353                                  <1>         ;add    esp, 20 ; --> EIP, CS, EFLAGS, ESP, SS
12354                                  <1>         ;xor    eax, eax ; 0
12355 00010DB9 FEC8                    <1>         dec     al ; eax = 0
12356                                  <1>         ;mov    edx, UDATA
12357                                  <1>         ; 18/11/2017
12358 00010DBB 6A23                    <1>         push    UDATA ; user's stack segment
12359                                  <1>         ;push   edx
12360 00010DBD 55                      <1>         push    ebp ; user's stack pointer
12361                                  <1>                     ; (points to number of arguments)
12362                                  <1>
12363                                  <1>         ; 04/01/2017
12364                                  <1>         ; MainProg comes here while [sysflg]= 0FFh
12365                                  <1>         ; (but sysexec comes here while [sysflg]= 0)
12366 00010DBE C605[5B030300]00        <1>         mov     byte [sysflg], 0 ; 04/01/2017
12367                                  <1>                                  ; (timer_int sysflg control)
12368 00010DC5 FB                      <1>         sti
12369 00010DC6 9C                      <1>         pushfd ; EFLAGS
12370                                  <1>                ; Set IF for enabling interrupts in user mode
12371                                  <1>         ;or     dword [esp], 200h
12372                                  <1>         ;
12373                                  <1>         ;mov    bx, UCODE
12374                                  <1>         ;push   bx ; user's code segment
12375 00010DC7 6A1B                    <1>         push    UCODE
12376                                  <1>         ;push   0
12377 00010DC9 50                      <1>         push    eax ; EIP (=0) - start address -
12378 00010DCA 8925[5C030300]          <1>         mov     [u.sp], esp ; 29/07/2015
12379                                  <1>         ; 05/08/2015
12380                                  <1>         ; Remedy of a General Protection Fault during 'iretd' is here !
12381                                  <1>         ; ('push dx' would cause to general protection fault,
12382                                  <1>         ; after 'pop ds' etc.)
12383                                  <1>         ;
12384                                  <1>         ;; push dx ; ds (UDATA)
12385                                  <1>         ;; push dx ; es (UDATA)
12386                                  <1>         ;; push dx ; fs (UDATA)
12387                                  <1>         ;; push dx ; gs (UDATA)
12388                                  <1>         ;
12389                                  <1>         ; This is a trick to prevent general protection fault
12390                                  <1>         ; during 'iretd' intruction at the end of 'sysrele' (in u1.s):
12391 00010DD0 66BA2300                <1>         mov     dx, UDATA ; 19/11/2017
12392 00010DD4 8EC2                    <1>         mov     es, dx ; UDATA
12393 00010DD6 06                      <1>         push    es ; ds (UDATA)
12394 00010DD7 06                      <1>         push    es ; es (UDATA)
12395 00010DD8 06                      <1>         push    es ; fs (UDATA)
12396 00010DD9 06                      <1>         push    es ; gs (UDATA)
12397 00010DDA 66BA1000                <1>         mov     dx, KDATA
12398 00010DDE 8EC2                    <1>         mov     es, dx
12399                                  <1>         ;
12400                                  <1>         ;; pushad simulation
12401 00010DE0 89E5                    <1>         mov     ebp, esp ; esp before pushad
12402 00010DE2 50                      <1>         push    eax ; eax (0)
12403 00010DE3 50                      <1>         push    eax ; ecx (0)
12404 00010DE4 50                      <1>         push    eax ; edx (0)
12405 00010DE5 50                      <1>         push    eax ; ebx (0)
12406 00010DE6 55                      <1>         push    ebp ; esp before pushad
12407 00010DE7 50                      <1>         push    eax ; ebp (0)
12408 00010DE8 50                      <1>         push    eax ; esi (0)
12409 00010DE9 50                      <1>         push    eax ; edi (0)
12410                                  <1>         ;
12411 00010DEA A3[64030300]            <1>         mov     [u.r0], eax ; eax = 0
12412 00010DEF 8925[60030300]          <1>         mov     [u.usp], esp
12413                                  <1>
12414                                  <1>         ; 14/11/2017
12415 00010DF5 E9F4CAFFFF              <1>         jmp     sysret0
12416                                  <1>
12417                                  <1> ;       ; 02/05/2016
12418                                  <1> ;       ;inc    byte [sysflg] ; 0FFh -> 0
12419                                  <1> ;       ;mov    byte [sysflg], 0 ; 04/01/2017
12420                                  <1> ;       movzx   ebx, byte [u.uno]
12421                                  <1> ;       shl     bl, 1 ; 13/11/2017
12422                                  <1> ;       cmp     word [ebx+p.ppid-2], 1 ; MainProg
12423                                  <1> ;       ja      sysret0 ; 03/05/2016
12424                                  <1> ;       push    sysret ; *
12425                                  <1> ;       mov     [u.usp], esp
12426                                  <1> ;       call    wswap ; save child process 'u' structure and
12427                                  <1> ;                     ; registers
12428                                  <1> ;       add     dword [u.usp], 4 ; 03/05/2016
12429                                  <1> ;sysexec_19: ; 02/05/2016
12430                                  <1> ;       retn ; * 'sysret' ; byte [sysflg] -> 0FFh
12431                                  <1>
12432                                  <1> readi:
12433                                  <1>         ; 01/05/2016
12434                                  <1>         ; 25/04/2016 - TRDOS 386 (TRDOS v2.0)
12435                                  <1>         ; 20/05/2015 - Retro UNIX 386 v1
12436                                  <1>         ; 11/03/2013 - 31/07/2013 (Retro UNIX 8086 v1)
12437                                  <1>         ;
12438                                  <1>         ; Reads from a file whose the first cluster number in EAX
12439                                  <1>         ;
12440                                  <1>         ; INPUTS ->
12441                                  <1>         ;    EAX - First cluster number of the file
```

```
12442                                  <1>       ;    u.count - byte count user desires
12443                                  <1>       ;    u.base - points to user buffer
12444                                  <1>       ;    u.fofp - points to dword with current file offset
12445                                  <1>       ;    i.size - file size
12446                                  <1>       ;    cdev - logical dos drive number of the file
12447                                  <1>       ; OUTPUTS ->
12448                                  <1>       ;    u.count - cleared
12449                                  <1>       ;    u.nread - accumulates total bytes passed back
12450                                  <1>       ;
12451                                  <1>       ; ((EAX)) input/output
12452                                  <1>       ; (Retro UNIX Prototype : 14/12/2012 - 01/03/2013, UNIXCOPY.ASM)
12453                                  <1>         ; ((Modified registers: edx, ebx, ecx, esi, edi))
12454                                  <1>
12455 00010DFA 31D2                   <1>       xor    edx, edx ; 0
12456 00010DFC 8915[8C030300]         <1>       mov    [u.nread], edx ; 0
12457 00010E02 668915[C4030300]       <1>       mov    [u.pcount], dx ; 19/05/2015
12458 00010E09 3915[88030300]         <1>       cmp    [u.count], edx ; 0
12459 00010E0F 7701                   <1>       ja     short readi_1
12460 00010E11 C3                     <1>       retn
12461                                  <1> readi_1:
12462                                  <1> dskr:
12463                                  <1>       ; 01/05/2016
12464                                  <1>       ; 25/04/2016 - TRDOS 386 (TRDOS v2.0)
12465                                  <1>       ; 24/05/2015 - 12/10/2015 (Retro UNIX 386 v1)
12466                                  <1>       ; 26/04/2013 - 03/08/2013 (Retro UNIX 8086 v1)
12467                                  <1> dskr_0:
12468 00010E12 8B15[55040300]         <1>         mov  edx, [i.size]
12469 00010E18 8B1D[74030300]         <1>       mov    ebx, [u.fofp]
12470 00010E1E 2B13                   <1>       sub    edx, [ebx]
12471 00010E20 7647                   <1>       jna    short dskr_4
12472                                  <1>       ;
12473 00010E22 50                     <1>       push   eax ; 01/05/2016
12474 00010E23 3B15[88030300]         <1>       cmp     edx, [u.count]
12475 00010E29 7306                   <1>       jnb    short dskr_1
12476 00010E2B 8915[88030300]         <1>       mov    [u.count], edx
12477                                  <1> dskr_1:
12478                                  <1>       ; EAX = First Cluster
12479                                  <1>       ; [Current_Drv] = Physical drive number
12480 00010E31 E83B000000             <1>       call   mget_r
12481                                  <1>       ; NOTE: in 'mget_r', relevant sector will be read in buffer
12482                                  <1>       ; if it is not already in buffer !
12483 00010E36 BB[8C050300]           <1>       mov    ebx, readi_buffer
12484 00010E3B 803D[C6030300]00       <1>       cmp    byte [u.kcall], 0 ; the caller is 'namei' sign (=1)
12485 00010E42 770F                   <1>       ja     short dskr_3   ; zf=0 -> the caller is 'namei'
12486 00010E44 66833D[C4030300]00     <1>       cmp    word [u.pcount], 0
12487 00010E4C 7705                   <1>       ja     short dskr_3
12488                                  <1> dskr_2:
12489                                  <1>       ; [u.base] = virtual address to transfer (as destination address)
12490 00010E4E E894010000             <1>       call   trans_addr_w ; translate virtual address to physical (w)
12491                                  <1> dskr_3:
12492                                  <1>       ; EBX (r5) = system (I/O) buffer address -physical-
12493 00010E53 E8F7010000             <1>       call   sioreg
12494 00010E58 87F7                   <1>       xchg   esi, edi
12495                                  <1>       ; EDI = file (user data) offset
12496                                  <1>       ; ESI = sector (I/O) buffer offset
12497                                  <1>       ; ECX = byte count
12498 00010E5A F3A4                   <1>       rep    movsb
12499                                  <1>       ; eax = remain bytes in buffer
12500                                  <1>         ;       (check if remain bytes in the buffer > [u.pcount])
12501 00010E5C 09C0                   <1>       or     eax, eax
12502 00010E5E 75EE                   <1>       jnz    short dskr_2 ; (page end before system buffer end!)
12503 00010E60 58                     <1>       pop    eax  ; (first cluster number)
12504 00010E61 390D[88030300]         <1>       cmp    [u.count], ecx ; 0
12505 00010E67 77A9                   <1>       ja     short dskr_0
12506                                  <1> dskr_4:
12507 00010E69 C605[C6030300]00       <1>       mov    byte [u.kcall], 0
12508 00010E70 C3                     <1>       retn
12509                                  <1>
12510                                  <1> mget_r:
12511                                  <1>       ; 24/10/2016
12512                                  <1>       ; 22/10/2016
12513                                  <1>       ; 12/10/2016
12514                                  <1>       ; 29/04/2016
12515                                  <1>       ; 25/04/2016 - TRDOS 386 (TRDOS v2.0)
12516                                  <1>       ; 03/06/2015 (Retro UNIX 386 v1, 'mget', u.5s)
12517                                  <1>       ; 22/03/2013 - 31/07/2013 (Retro UNIX 8086 v1)
12518                                  <1>       ;
12519                                  <1>       ; Get existing or (allocate) a new disk block for file
12520                                  <1>       ;
12521                                  <1>       ; INPUTS ->
12522                                  <1>       ;    [u.fofp] = file offset pointer
12523                                  <1>       ;    EAX = First Cluster
12524                                  <1>       ;    [cdev] = Logical dos drive number
12525                                  <1>       ;    ([u.off] = file offset)
12526                                  <1>       ; OUTPUTS ->
12527                                  <1>       ;    EAX = logical sector number
12528                                  <1>       ;    ESI = Logical Dos Drive Description Table address
12529                                  <1>       ;
12530                                  <1>       ; Modified registers: EDX, EBX, ECX, ESI, EDI
12531                                  <1>
12532 00010E71 8B35[74030300]         <1>       mov     esi, [u.fofp]
12533 00010E77 8B1E                   <1>       mov    ebx, [esi] ; (u.off)
12534                                  <1>
12535 00010E79 29C9                   <1>       sub    ecx, ecx
12536 00010E7B 8A2D[46030300]         <1>       mov    ch, [cdev]
12537                                  <1>
12538 00010E81 BE00010900             <1>       mov    esi, Logical_DOSDisks
12539 00010E86 01CE                   <1>       add    esi, ecx
12540                                  <1>
12541 00010E88 380D[C48D0100]         <1>       cmp    [readi.valid], cl ; 0
12542 00010E8E 7649                   <1>       jna    short mget_r_0
12543                                  <1>
12544 00010E90 3A2D[C58D0100]         <1>       cmp    ch, [readi.drv]
12545 00010E96 7541                   <1>       jne    short mget_r_0
12546                                  <1>
```

```
12547 00010E98 3B05[D88D0100]      <1>       cmp    eax, [readi.fclust]
12548 00010E9E 7565                <1>       jne    short mget_r_3
12549                              <1>
12550 00010EA0 89D8                <1>       mov    eax, ebx ; file offset
12551 00010EA2 668B0D[CC8D0100]    <1>       mov    cx, [readi.bpc]
12552 00010EA9 41                  <1>       inc    ecx ; <= 65536
12553 00010EAA 29D2                <1>       sub    edx, edx
12554 00010EAC F7F1                <1>       div    ecx
12555                              <1>
12556 00010EAE 8B3D[D48D0100]      <1>       mov    edi, [readi.c_index] ; cluster index
12557                              <1>
12558 00010EB4 39F8                <1>       cmp    eax, edi
12559 00010EB6 757A                <1>         jne     short mget_r_4  ; (*)
12560                              <1>
12561                              <1>       ; edx = byte offset in cluster (<= 65535)
12562 00010EB8 668915[CE8D0100]    <1>       mov    [readi.offset], dx
12563 00010EBF 66C1EA09            <1>       shr    dx, 9 ; / 512
12564 00010EC3 8815[C78D0100]      <1>       mov    [readi.s_index], dl ; sector index in cluster (0 to spc -1)
12565                              <1>
12566 00010EC9 A1[D08D0100]        <1>       mov    eax, [readi.cluster]  ; > 0 if [readi.valid] = 1
12567 00010ECE 8B15[DC8D0100]      <1>       mov    edx, [readi.fs_index]
12568 00010ED4 E99A000000          <1>         jmp     mget_r_7
12569                              <1>
12570                              <1> mget_r_0:
12571 00010ED9 882D[C58D0100]      <1>       mov    [readi.drv], ch ; physical drive number
12572 00010EDF 807E0300            <1>       cmp    byte [esi+LD_FATType], 0
12573 00010EE3 7707                <1>       ja     short mget_r_1
12574 00010EE5 8A4E12              <1>       mov    cl, [esi+LD_FS_BytesPerSec+1]
12575 00010EE8 D0E9                <1>       shr    cl, 1 ;  ; 1 for 512 bytes, 4 for 2048 bytes
12576 00010EEA EB03                <1>       jmp    short mget_r_2
12577                              <1> mget_r_1:
12578 00010EEC 8A4E13              <1>       mov    cl, [esi+LD_BPB+BPB_SecPerClust]
12579                              <1> mget_r_2:
12580 00010EEF 880D[C68D0100]      <1>       mov    [readi.spc], cl  ; sectors per cluster
12581                              <1>       ; NOTE: readi bytes per sector value is always 512 !
12582 00010EF5 66C1E109            <1>       shl    cx, 9 ; * 512
12583 00010EF9 6649                <1>       dec    cx ; bytes per cluster - 1
12584 00010EFB 66890D[CC8D0100]    <1>       mov    [readi.bpc], cx
12585 00010F02 6629C9              <1>       sub    cx, cx
12586                              <1> mget_r_3:
12587 00010F05 A3[D88D0100]        <1>       mov    [readi.fclust], eax ; first cluster (or FDT address)
12588 00010F0A 880D[C48D0100]      <1>       mov    [readi.valid], cl ; 0
12589                              <1>       ;mov   [readi.s_index], cl ; 0
12590                              <1>       ;mov   [readi.offset], cx ; 0
12591 00010F10 890D[D48D0100]      <1>       mov    [readi.c_index], ecx ; 0
12592 00010F16 890D[D08D0100]      <1>       mov    [readi.cluster], ecx ; 0
12593 00010F1C 890D[C88D0100]      <1>       mov    [readi.sector], ecx ; 0
12594                              <1>
12595 00010F22 89D8                <1>       mov    eax, ebx ; file offset
12596 00010F24 668B0D[CC8D0100]    <1>       mov    cx, [readi.bpc]
12597 00010F2B 41                  <1>       inc    ecx ; <= 65536
12598 00010F2C 29D2                <1>       sub    edx, edx
12599 00010F2E F7F1                <1>       div    ecx
12600                              <1>       ;mov   edi, [readi.c_index] ; previous cluster index
12601 00010F30 29FF                <1>       sub    edi, edi
12602                              <1> mget_r_4:
12603 00010F32 A3[D48D0100]        <1>       mov    [readi.c_index], eax ; cluster index
12604                              <1>       ; edx = byte offset in cluster (<= 65535)
12605 00010F37 668915[CE8D0100]    <1>       mov    [readi.offset], dx
12606 00010F3E 66C1EA09            <1>       shr    dx, 9 ; / 512
12607 00010F42 8815[C78D0100]      <1>       mov    [readi.s_index], dl ; sector index in cluster (0 to spc -1)
12608                              <1>
12609 00010F48 89C1                <1>       mov    ecx, eax ; current cluster index
12610 00010F4A A1[D88D0100]        <1>       mov    eax, [readi.fclust]
12611 00010F4F 09C9                <1>       or     ecx, ecx ; cluster index
12612 00010F51 741B                <1>       jz     short mget_r_6
12613                              <1>
12614 00010F53 39CF                <1>       cmp    edi, ecx
12615 00010F55 7710                <1>       ja     short mget_r_5 ; old cluster index is higher
12616 00010F57 8B15[D08D0100]      <1>       mov    edx, [readi.cluster]
12617 00010F5D 21D2                <1>       and    edx, edx
12618 00010F5F 7406                <1>       jz     short mget_r_5
12619                              <1>       ; valid 'readi' parameters (*)
12620 00010F61 89D0                <1>       mov    eax, edx
12621 00010F63 29F9                <1>       sub    ecx, edi
12622 00010F65 740C                <1>       jz     short mget_r_7
12623                              <1> mget_r_5:
12624                              <1>       ; EAX = Beginning cluster
12625                              <1>       ; EDX = Sector index in disk/file section
12626                              <1>       ;      (Only for SINGLIX file system!)
12627                              <1>       ; ECX = Cluster sequence number after the beginning cluster
12628                              <1>       ; ESI = Logical DOS Drive Description Table address
12629 00010F67 E8F9C7FFFF          <1>       call   get_cluster_by_index
12630 00010F6C 724E                <1>       jc     short mget_r_err
12631                              <1>       ; EAX = Cluster number
12632                              <1> mget_r_6:
12633 00010F6E A3[D08D0100]        <1>       mov    [readi.cluster], eax ; FDT number for Singlix File System
12634                              <1> mget_r_7:
12635 00010F73 807E0300            <1>       cmp    byte [esi+LD_FATType], 0
12636 00010F77 765F                <1>       jna    short mget_r_12
12637                              <1>
12638 00010F79 83E802              <1>       sub    eax, 2
12639 00010F7C 0FB615[C68D0100]    <1>       movzx  edx, byte [readi.spc]
12640 00010F83 F7E2                <1>       mul    edx
12641                              <1>
12642 00010F85 034668              <1>       add    eax, [esi+LD_DATABegin]
12643 00010F88 8A15[C78D0100]      <1>       mov    dl, [readi.s_index]
12644 00010F8E 01D0                <1>       add    eax, edx
12645                              <1> mget_r_8:
12646                              <1>       ; eax = logical sector number
12647 00010F90 803D[C48D0100]00    <1>       cmp    byte [readi.valid], 0
12648 00010F97 7608                <1>       jna    short mget_r_9
12649 00010F99 3B05[C88D0100]      <1>       cmp    eax, [readi.sector]
12650 00010F9F 7436                <1>       je     short mget_r_11 ; sector is already in 'readi' buffer
12651                              <1> mget_r_9:
```

```
12652 00010FA1 A3[C88D0100]        <1>       mov   [readi.sector], eax
12653 00010FA6 BB[8C050300]        <1>       mov   ebx, readi_buffer ; buffer address
12654 00010FAB B901000000          <1>       mov   ecx, 1
12655                              <1>       ; 29/04/2016
12656                              <1>       ;xor  dl, dl
12657                              <1>
12658                              <1>       ; EAX = Logical sector number
12659                              <1>       ; ECX = Sector count
12660                              <1>       ; EBX = Buffer address
12661                              <1>       ; (EDX = 0)
12662                              <1>       ; ESI = Logical DOS drive description table address
12663                              <1>
12664 00010FB0 E868130000          <1>       call  disk_read
12665 00010FB5 7314                <1>       jnc   short mget_r_10
12666                              <1>
12667                              <1>       ; 22/10/2016 (15h -> 17)
12668 00010FB7 B811000000          <1>       mov   eax, 17 ; Drive not ready or read error !
12669                              <1> mget_r_err:
12670 00010FBC A3[C8030300]        <1>       mov   [u.error], eax
12671                              <1>       ; 12/10/2016
12672 00010FC1 A3[64030300]        <1>       mov   [u.r0], eax
12673 00010FC6 E901C9FFFF          <1>       jmp   error
12674                              <1> mget_r_10:
12675 00010FCB C605[C48D0100]01    <1>       mov   byte [readi.valid], 1 ; 24/10/2016
12676 00010FD2 A1[C88D0100]        <1>       mov   eax, [readi.sector]
12677                              <1> mget_r_11:
12678 00010FD7 C3                  <1>       retn
12679                              <1> mget_r_12:
12680                              <1>       ; EAX = FDT number
12681                              <1>       ; EDX = Sector index from FDT sector (0,1,2,3,4...)
12682 00010FD8 40                  <1>       inc   eax ; the first data sector in FS disk section
12683 00010FD9 8915[DC8D0100]      <1>       mov   [readi.fs_index], edx
12684 00010FDF 01D0                <1>       add   eax, edx
12685 00010FE1 EBAD                <1>       jmp   short mget_r_8
12686                              <1>
12687                              <1> trans_addr_r:
12688                              <1>       ; 12/10/2016
12689                              <1>       ; 02/05/2016 - TRDOS 386 (TRDOS v2.0)
12690                              <1>       ; Translate virtual address to physical address
12691                              <1>       ; for reading from user's memory space
12692                              <1>       ; 04/06/2015 - 18/10/2015 (Retro UNIX 386 v1)
12693                              <1>
12694 00010FE3 31D2                <1>       xor   edx, edx ; 0 (read access sign)
12695 00010FE5 EB04                <1>       jmp   short trans_addr_rw
12696                              <1>
12697                              <1> trans_addr_w:
12698                              <1>       ; 12/10/2016
12699                              <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
12700                              <1>       ; Translate virtual address to physical address
12701                              <1>       ; for writing to user's memory space
12702                              <1>       ; 04/06/2015 - 18/10/2015 (Retro UNIX 386 v1)
12703                              <1>
12704 00010FE7 29D2                <1>       sub   edx, edx
12705 00010FE9 FEC2                <1>       inc   dl ; 1 (write access sign)
12706                              <1> trans_addr_rw:
12707 00010FEB 50                  <1>       push  eax
12708 00010FEC 53                  <1>       push  ebx
12709 00010FED 52                  <1>       push  edx ; r/w sign (in DL)
12710                              <1>       ;
12711 00010FEE 8B1D[84030300]      <1>       mov   ebx, [u.base]
12712 00010FF4 E8E552FFFF          <1>       call  get_physical_addr ; get physical address
12713 00010FF9 730F                <1>       jnc   short passc_0
12714 00010FFB A3[C8030300]        <1>       mov   [u.error], eax
12715 00011000 A3[64030300]        <1>       mov   [u.r0], eax ; 12/10/2016
12716                              <1>       ;pop  edx
12717                              <1>       ;pop  ebx
12718                              <1>       ;pop  eax
12719 00011005 E9C2C8FFFF          <1>       jmp   error
12720                              <1> passc_0:
12721 0001100A F6C202              <1>       test  dl, PTE_A_WRITE ; writable page
12722 0001100D 5A                  <1>       pop   edx
12723 0001100E 751C                <1>       jnz   short passc_1
12724                              <1>
12725 00011010 20D2                <1>       and   dl, dl
12726 00011012 7418                <1>       jz    short passc_1
12727                              <1>       ; read only (duplicated) page -must be copied to a new page-
12728                              <1>       ; EBX = linear address
12729 00011014 51                  <1>       push  ecx
12730 00011015 E85D4FFFFF          <1>       call  copy_page
12731 0001101A 59                  <1>       pop   ecx
12732 0001101B 721E                <1>       jc    short passc_2
12733 0001101D 50                  <1>       push  eax ; physical address of the new/allocated page
12734 0001101E E8E551FFFF          <1>       call  add_to_swap_queue
12735 00011023 58                  <1>       pop   eax
12736 00011024 81E3FF0F0000        <1>       and   ebx, PAGE_OFF ; 0FFFh
12737                              <1>       ;mov  ecx, PAGE_SIZE
12738                              <1>       ;sub  ecx, ebx
12739 0001102A 01D8                <1>       add   eax, ebx
12740                              <1> passc_1:
12741 0001102C A3[C0030300]        <1>       mov   [u.pbase], eax ; physical address
12742 00011031 66890D[C4030300]    <1>       mov   [u.pcount], cx ; remain byte count in page (1-4096)
12743 00011038 5B                  <1>       pop   ebx
12744 00011039 58                  <1>       pop   eax
12745 0001103A C3                  <1>       retn
12746                              <1> passc_2:
12747 0001103B B804000000          <1>       mov   eax, ERR_MINOR_IM ; "Insufficient memory !" error
12748 00011040 A3[64030300]        <1>       mov   [u.r0], eax ; 12/10/2016
12749 00011045 A3[C8030300]        <1>       mov   dword [u.error], eax
12750                              <1>       ;pop  ebx
12751                              <1>       ;pop  eax
12752 0001104A E97DC8FFFF          <1>       jmp   error
12753                              <1>
12754                              <1> sioreg:
12755                              <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
12756                              <1>       ; 19/05/2015 - 25/07/2015 (Retro UNIX 386 v1)
```

```
12757                                  <1>       ; 12/03/2013 - 22/07/2013 (Retro UNIX 8086 v1)
12758                                  <1>       ; INPUTS ->
12759                                  <1>       ;     EBX = system buffer (data) address (r5)
12760                                  <1>       ;     [u.fofp] = pointer to file offset pointer
12761                                  <1>       ;     [u.base] = virtual address of the user buffer
12762                                  <1>       ;     [u.pbase] = physical address of the user buffer
12763                                  <1>       ;     [u.count] = byte count
12764                                  <1>       ;     [u.pcount] = byte count within page frame
12765                                  <1>       ; OUTPUTS ->
12766                                  <1>       ;     ESI = user data offset (r1)
12767                                  <1>       ;     EDI = system (I/O) buffer offset (r2)
12768                                  <1>       ;     ECX = byte count (r3)
12769                                  <1>       ;     EAX = remain bytes after byte count within page frame
12770                                  <1>       ;      (If EAX > 0, transfer will continue from the next page)
12771                                  <1>         ;
12772                                  <1>       ; ((Modified registers:  EDX))
12773                                  <1>
12774 0001104F 8B35[74030300]           <1>         mov     esi, [u.fofp]
12775 00011055 8B3E                    <1>         mov     edi, [esi]
12776 00011057 89F9                    <1>       mov   ecx, edi
12777 00011059 81C900FEFFFF            <1>       or    ecx, 0FFFFFE00h
12778 0001105F 81E7FF010000            <1>       and   edi, 1FFh
12779 00011065 01DF                    <1>       add   edi, ebx ; EBX = system buffer (data) address
12780 00011067 F7D9                    <1>       neg   ecx
12781 00011069 3B0D[88030300]          <1>       cmp   ecx, [u.count]
12782 0001106F 7606                    <1>       jna   short sioreg_0
12783 00011071 8B0D[88030300]          <1>       mov   ecx, [u.count]
12784                                  <1> sioreg_0:
12785 00011077 803D[C6030300]00        <1>       cmp   byte [u.kcall], 0
12786 0001107E 7613                    <1>       jna   short sioreg_1
12787                                  <1>        ; the caller is 'mkdir' or 'namei'
12788 00011080 A1[84030300]            <1>       mov   eax, [u.base]
12789 00011085 A3[C0030300]            <1>       mov   [u.pbase], eax ; physical address = virtual address
12790 0001108A 66890D[C4030300]        <1>       mov   word [u.pcount], cx ; remain bytes in buffer (1 sector)
12791 00011091 EB0B                    <1>       jmp   short sioreg_2
12792                                  <1> sioreg_1:
12793 00011093 0FB715[C4030300]        <1>       movzx  edx, word [u.pcount]
12794 0001109A 39D1                    <1>       cmp   ecx, edx
12795 0001109C 772A                    <1>       ja    short sioreg_4 ; transfer count > [u.pcount]
12796                                  <1> sioreg_2: ; 2:
12797 0001109E 31C0                    <1>       xor   eax, eax
12798                                  <1> sioreg_3:
12799 000110A0 010D[8C030300]          <1>       add   [u.nread], ecx
12800 000110A6 290D[88030300]          <1>       sub   [u.count], ecx
12801 000110AC 010D[84030300]          <1>       add   [u.base], ecx
12802 000110B2 010E                    <1>         add [esi], ecx
12803 000110B4 8B35[C0030300]          <1>       mov   esi, [u.pbase]
12804 000110BA 66290D[C4030300]        <1>       sub   [u.pcount], cx
12805 000110C1 010D[C0030300]          <1>       add   [u.pbase], ecx
12806 000110C7 C3                      <1>         retn
12807                                  <1> sioreg_4:
12808                                  <1>       ; transfer count > [u.pcount]
12809                                  <1>       ; (ecx > edx)
12810 000110C8 89C8                    <1>       mov   eax, ecx
12811 000110CA 29D0                    <1>       sub   eax, edx ; remain bytes for 1 sector (block) transfer
12812 000110CC 89D1                    <1>       mov   ecx, edx ; current transfer count = [u.pcount]
12813 000110CE EBD0                    <1>       jmp   short sioreg_3
12814                                  <1>
12815                                  <1> tswitch: ; Retro UNIX 386 v1
12816                                  <1> tswap:
12817                                  <1>       ; 16/01/2017
12818                                  <1>       ; 21/05/2016 - TRDOS 386 (TRDOS v2.0)
12819                                  <1>       ; 10/05/2015 - 01/09/2015 (Retro UNIX 386 v1)
12820                                  <1>       ; 14/04/2013 - 14/02/2014 (Retro UNIX 8086 v1)
12821                                  <1>       ; time out swap, called when a user times out.
12822                                  <1>       ; the user is put on the low priority queue.
12823                                  <1>       ; This is done by making a link from the last user
12824                                  <1>       ; on the low priority queue to him via a call to 'putlu'.
12825                                  <1>       ; then he is swapped out.
12826                                  <1>
12827                                  <1>       ; TRDOS 386 (TRDOS v2.0) modification ->  ** 21/05/2016 **
12828                                  <1>       ;     * when a high priority (event) process will be stopped
12829                                  <1>       ;     (swapped out, swithched out/off), 'tswap/tswitch' will
12830                                  <1>       ;     not add it to a run queue.
12831                                  <1>       ;     /// What for: Process may be already in a run queue,
12832                                  <1>       ;     it is unspeficied state because process might be started
12833                                  <1>       ;     by a timer event which does not regard previous priority
12834                                  <1>       ;     level and run queue of the process (for fast executing!).
12835                                  <1>       ;     After the 'run for event', process will be sequenced
12836                                  <1>       ;     to run by it's actual run queue. ///
12837                                  <1>       ;
12838                                  <1>       ; Retro UNIX 386 v1 modification ->
12839                                  <1>       ;       swap (software task switch) is performed by changing
12840                                  <1>       ;     user's page directory (u.pgdir) instead of segment change
12841                                  <1>       ;     as in Retro UNIX 8086 v1.
12842                                  <1>       ;
12843                                  <1>       ; RETRO UNIX 8086 v1 modification ->
12844                                  <1>       ;       'swap to disk' is replaced with 'change running segment'
12845                                  <1>       ;     according to 8086 cpu (x86 real mode) architecture.
12846                                  <1>       ;     pdp-11 was using 64KB uniform memory while IBM PC
12847                                  <1>       ;     compatibles was using 1MB segmented memory
12848                                  <1>       ;     in 8086/8088 times.
12849                                  <1>       ;
12850                                  <1>       ; INPUTS ->
12851                                  <1>       ;    u.uno - users process number
12852                                  <1>       ;    runq+4 - lowest priority queue
12853                                  <1>       ; OUTPUTS ->
12854                                  <1>       ;    r0 - users process number
12855                                  <1>       ;    r2 - lowest priority queue address
12856                                  <1>       ;
12857                                  <1>       ; ((AX = R0, BX = R2)) output
12858                                  <1>       ; ((Modified registers: EDX, EBX, ECX, ESI, EDI))
12859                                  <1>       ;
12860                                  <1>
12861                                  <1>       NOTE:
```

```
12862                                  <1>       ;* [u.pri] priority level is specified by run queue which is process
12863                                  <1>       ;  comes to run from.
12864                                  <1>       ;* Initial [u.pri] is 1 ('normal/regular') for programs
12865                                  <1>       ;  (which are launched by MainProg or 'sysexec'), it is changed
12866                                  <1>       ;  to 2 ('high') by timer event, if program uses 'systimer' system call.
12867                                  <1>       ;* Program (Process) also can change it's running priority
12868                                  <1>       ;  from 1 to 0 or up to 2 by using 'syspri' system call; but,
12869                                  <1>       ;  if program selects priority level 2 (high) for running, next time
12870                                  <1>       ;  it is reduced to 1 (normal/regular) because 'syspri' adds this
12871                                  <1>       ;  program to 'run for normal' queue while running duration is a bit
12872                                  <1>       ;  protected from swap/switch out immediate, behalf of other high
12873                                  <1>       ;  priority process in sequence. Program (with high priority) will not
12874                                  <1>       ;  be swapped/switched out (by timer event) before it's time quantum
12875                                  <1>       ;  will be elapsed, but, this will be temporary if program is not using
12876                                  <1>       ;  timer event function.
12877                                  <1>
12878                                  <1>       ;For example:
12879                                  <1>       ;If a process frequently gets a timer event, it runs at high priority
12880                                  <1>       ;level but when it returns from running it returns to actual run queue,
12881                                  <1>       ;not to 'run for event' queue again.
12882                                  <1>       ;'tswap' will not change the sequence at return/stop(swap out) stage.
12883                                  <1>       ;But if priority level not high (=2, 'run for event'), 'tswap/tswitch'
12884                                  <1>       ;will add the stopping process to relevant run queue according to
12885                                  <1>       ;[u.pri] priority level.
12886                                  <1>
12887                                  <1>       ; 16/01/2017
12888 000110D0 BB[54030300]            <1>       mov    ebx, runq+2  ; 'runq_normal' ; normal/regular priority
12889                                  <1>       ; 21/05/2016
12890                                  <1>       ;cmp   byte [u.pri], 2     ; high priority (run for event) ?
12891                                  <1>       ;jnb   short swap
12892                                  <1>       ; 16/01/2017
12893                                  <1>       ; (Normal and also high/event priority processes will be added to
12894                                  <1>       ; normal priority run queue for ensuring circular running sequence!)
12895                                  <1>       ; (Timer interrupt or 'syspri' system call may change priority and run
12896                                  <1>       ; queue to high/event level.)
12897 000110D5 803D[A9030300]00       <1>       cmp    byte [u.pri], 0
12898 000110DC 7702                  <1>       ja     short tswap_1; normal priority run queue
12899                                  <1>       ;
12900 000110DE 43                    <1>       inc    ebx
12901 000110DF 43                    <1>       inc    ebx          ; runq+4, 'runq_background', low priority
12902                                  <1> tswap_1:
12903 000110E0 A0[B3030300]          <1>       mov    al, [u.uno]
12904                                  <1>              ; movb u.uno,r1 / move users process number to r1
12905                                  <1>       ; mov  $runq+4,r2
12906                                  <1>              ; / move lowest priority queue address to r2
12907                                  <1>       ; ebx = run queue
12908 000110E5 E8FE000000            <1>       call   putlu
12909                                  <1>              ; jsr r0,putlu / create link from last user on Q to
12910                                  <1>                       ; / u.uno's user
12911                                  <1>
12912                                  <1> switch: ; Retro UNIX 386 v1
12913                                  <1> swap:
12914                                  <1>       ; 02/01/2017
12915                                  <1>       ; 21/05/2016
12916                                  <1>       ; 20/05/2016
12917                                  <1>       ; 02/05/2016
12918                                  <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
12919                                  <1>       ; 10/05/2015 - 02/09/2015 (Retro UNIX 386 v1)
12920                                  <1>       ; 14/04/2013 - 08/03/2014 (Retro UNIX 8086 v1)
12921                                  <1>       ;
12922                                  <1>       ; 'swap' is routine that controls the swapping of processes
12923                                  <1>       ; in and out of core.
12924                                  <1>       ;
12925                                  <1>       ; TRDOS 386 (TRDOS v2.0) modification ->  ** 20/05/2016 **
12926                                  <1>       ;     * 3 different priority level is applied
12927                                  <1>       ;      (just as original unix v1)
12928                                  <1>       ;      1) high priority (event) run queue, 'runq_event'
12929                                  <1>       ;      2) normal priority (regular) run queue, 'runq_normal'
12930                                  <1>       ;      3) low priority (background) run queue, 'runq_backgroud'
12931                                  <1>       ;      'swap' code will run a process which has max. priority
12932                                  <1>         ;       (for earliest event at first)
12933                                  <1>       ;
12934                                  <1>       ; Retro UNIX 386 v1 modification ->
12935                                  <1>       ;       swap (software task switch) is performed by changing
12936                                  <1>       ;     user's page directory (u.pgdir) instead of segment change
12937                                  <1>       ;     as in Retro UNIX 8086 v1.
12938                                  <1>       ;
12939                                  <1>       ; RETRO UNIX 8086 v1 modification ->
12940                                  <1>       ;       'swap to disk' is replaced with 'change running segment'
12941                                  <1>       ;     according to 8086 cpu (x86 real mode) architecture.
12942                                  <1>       ;     pdp-11 was using 64KB uniform memory while IBM PC
12943                                  <1>       ;     compatibles was using 1MB segmented memory
12944                                  <1>       ;     in 8086/8088 times.
12945                                  <1>       ;
12946                                  <1>       ; INPUTS ->
12947                                  <1>       ;    runq table - contains processes to run.
12948                                  <1>       ;    p.link - contains next process in line to be run.
12949                                  <1>       ;    u.uno - process number of process in core
12950                                  <1>       ;    s.stack - swap stack used as an internal stack for swapping.
12951                                  <1>       ; OUTPUTS ->
12952                                  <1>       ;    (original unix v1 -> present process to its disk block)
12953                                  <1>       ;    (original unix v1 -> new process into core ->
12954                                  <1>       ;        Retro Unix 8086 v1 -> segment registers changed
12955                                  <1>       ;        for new process)
12956                                  <1>       ;    u.quant = 3 (Time quantum for a process)
12957                                  <1>       ;     ((INT 1Ch count down speed -> 18.2 times per second)
12958                                  <1>       ;    RETRO UNIX 8086 v1 will use INT 1Ch (18.2 times per second)
12959                                  <1>       ;      for now, it will swap the process if there is not
12960                                  <1>       ;      a keyboard event (keystroke) (Int 15h, function 4Fh)
12961                                  <1>       ;      or will count down from 3 to 0 even if there is a
12962                                  <1>       ;       keyboard event locking due to repetitive key strokes.
12963                                  <1>       ;      u.quant will be reset to 3 for RETRO UNIX 8086 v1.
12964                                  <1>       ;
12965                                  <1>       ; ((Modified registers: EAX, EDX, EBX, ECX, ESI, EDI))
12966                                  <1>
```

```
12967                                  <1>       ;NOTE:
12968                                  <1>       ;High priority queue is the first for selecting a process to run.
12969                                  <1>       ;If there is not a process in high priority level run queue,
12970                                  <1>       ;a process in normal priority run queue will be selected
12971                                  <1>       ;or a proces in low priority run queue will be selected if normal
12972                                  <1>       ;priority level run queue is empty.
12973                                  <1>
12974                                  <1>       ; 21/05/2016 -(3 priority levels, 3 run queues)
12975 000110EA BE[52030300]            <1>       mov    esi, runq ; 'runq_event' ; high priority, 'run for event'
12976 000110EF C605[208E0100]03        <1>       mov    byte [priority], 3 ; high priority + 1
12977 000110F6 31DB                    <1>       xor    ebx, ebx ; 02/01/2017
12978                                  <1> swap_0: ; 1: / search runq table for highest priority process
12979 000110F8 66AD                    <1>       lodsw  ; mov ax, [esi], add esi+2
12980                                  <1>       ;xor   ebx, ebx ; 02/05/2016
12981 000110FA 6621C0                  <1>       and    ax, ax ; are there any processes to run in this Q entry
12982 000110FD 750E                    <1>       jnz    short swap_2
12983                                  <1>       ; 21/05/2026
12984                                  <1>       ; runq_normal = runq+2, runq_background = runq+4
12985 000110FF FE0D[208E0100]          <1>       dec    byte [priority] ; 3 -> 3, 2 -> 1, 1-> 0
12986 00011105 75F1                    <1>       jnz    short swap_0
12987                                  <1>       ;cmp   esi, runq+6  ; if zero compare address to end of table
12988                                  <1>       ;jb    short swap_0 ; if not at end, go back
12989                                  <1> swap_1:
12990                                  <1>       ; 02/05/2016
12991                                  <1>       ; 29/04/2016 (TRDOS 386 = TRDOS v2.0)
12992                                  <1>       ; No user process to run...
12993                                  <1>       ; Run the kernel process... MainProg: Internal Command Interpreter
12994 00011107 FEC0                    <1>       inc    al ; mov al, 1  ; process number of MainProg
12995 00011109 FEC3                    <1>       inc    bl ; mov bl, al ; 1
12996 0001110B EB1E                    <1>       jmp    short swap_4
12997                                  <1> swap_2:
12998                                  <1>       ; 21/05/2016
12999 0001110D FE0D[208E0100]          <1>       dec    byte [priority] ; priority level of present user/process
13000                                  <1>                              ; 0, 1, 2
13001 00011113 4E                      <1>       dec    esi
13002 00011114 4E                      <1>         dec     esi
13003                                  <1>       ;
13004 00011115 88C3                    <1>       mov    bl, al
13005 00011117 38E0                    <1>       cmp    al, ah ; is there only 1 process in the queue to be run
13006 00011119 740A                    <1>       je     short swap_3 ; yes
13007 0001111B 8AA3[9F000300]          <1>       mov    ah, [ebx+p.link-1]
13008 00011121 8826                    <1>              mov    [esi], ah ; move next process in line into run queue
13009 00011123 EB06                    <1>       jmp    short swap_4
13010                                  <1> swap_3:
13011 00011125 6631D2                  <1>       xor    dx, dx
13012 00011128 668916                  <1>       mov    [esi], dx ; zero the entry; no processes on the Q
13013                                  <1> swap_4:
13014 0001112B 8A25[B3030300]          <1>       mov    ah, [u.uno]
13015 00011131 38C4                    <1>       cmp    ah, al ; is this process the same as the process in core?
13016 00011133 743B                    <1>              je     short swap_8 ; yes, don't have to swap
13017 00011135 08E4                    <1>       or     ah, ah  ; is the process # = 0
13018 00011137 740D                    <1>              jz     short swap_6 ; 'sysexit'
13019                                  <1>       ;cmp   ah, al ;is this process the same as the process in core?
13020                                  <1>              ;je    short swap_8 ; yes, don't have to swap
13021 00011139 8925[60030300]          <1>       mov    [u.usp], esp ; return  address for 'syswait' & 'sleep'
13022 0001113F E834000000              <1>       call   wswap   ; write out core to disk
13023 00011144 EB1C                    <1>       jmp    short swap_7
13024                                  <1> swap_6:
13025                                  <1>       ; Deallocate memory pages belong to the process
13026                                  <1>       ; which is being terminated.
13027                                  <1>       ; (Retro UNIX 386 v1 modification !)
13028                                  <1>       ;
13029 00011146 53                      <1>       push   ebx
13030 00011147 A1[B8030300]            <1>       mov    eax, [u.pgdir]  ; page directory of the process
13031 0001114C 8B1D[BC030300]          <1>       mov    ebx, [u.ppgdir] ; page directory of the parent process
13032 00011152 E8AB4BFFFF              <1>       call   deallocate_page_dir
13033 00011157 A1[B4030300]            <1>       mov    eax, [u.upage] ; 'user' structure page of the process
13034 0001115C E8464CFFFF              <1>       call   deallocate_page
13035 00011161 5B                      <1>       pop    ebx
13036                                  <1> swap_7:
13037 00011162 C0E302                  <1>       shl    bl, 2 ; * 4
13038 00011165 8B83[BC000300]          <1>       mov    eax, [ebx+p.upage-4] ; the 'u' page of the new process
13039 0001116B E840000000              <1>       call   rswap ; read new process into core
13040                                  <1> swap_8:
13041                                  <1>       ; Retro UNIX  8086 v1 modification !
13042 00011170 C605[A8030300]04        <1>       mov    byte [u.quant], time_count
13043 00011177 C3                      <1>       retn
13044                                  <1>
13045                                  <1> wswap:  ; < swap out, swap to disk >
13046                                  <1>       ; 28/02/2017 (fnsave)
13047                                  <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
13048                                  <1>       ; 09/05/2015 (Retro UNIX 386 v1)
13049                                  <1>       ; 26/05/2013 - 08/03/2014 (Retro UNIX 8086 v1)
13050                                  <1>       ; 'wswap' writes out the process that is in core onto its
13051                                  <1>       ; appropriate disk area.
13052                                  <1>       ;
13053                                  <1>       ; Retro UNIX 386 v1 modification ->
13054                                  <1>       ;       User (u) structure content and the user's register content
13055                                  <1>       ;     will be copied to the process's/user's UPAGE (a page for
13056                                  <1>       ;     saving 'u' structure and user registers for task switching).
13057                                  <1>       ;     u.usp - points to kernel stack address which contains
13058                                  <1>       ;             user's registers while entering system call.
13059                                  <1>       ;     u.sp  - points to kernel stack address
13060                                  <1>       ;             to return from system call -for IRET-.
13061                                  <1>       ;     [u.usp]+32+16 = [u.sp]
13062                                  <1>       ;     [u.usp] -> edi, esi, ebp, esp (= [u.usp]+32), ebx,
13063                                  <1>       ;             edx, ecx, eax, gs, fs, es, ds, -> [u.sp].
13064                                  <1>       ;
13065                                  <1>       ; Retro UNIX 8086 v1 modification ->
13066                                  <1>       ;       'swap to disk' is replaced with 'change running segment'
13067                                  <1>       ;     according to 8086 cpu (x86 real mode) architecture.
13068                                  <1>       ;     pdp-11 was using 64KB uniform memory while IBM PC
13069                                  <1>       ;     compatibles was using 1MB segmented memory
13070                                  <1>       ;     in 8086/8088 times.
13071                                  <1>       ;
```

```
13072                                  <1>       ; INPUTS ->
13073                                  <1>       ;    u.break - points to end of program
13074                                  <1>       ;    u.usp - stack pointer at the moment of swap
13075                                  <1>       ;    core - beginning of process program
13076                                  <1>       ;    ecore - end of core
13077                                  <1>       ;    user - start of user parameter area
13078                                  <1>       ;    u.uno - user process number
13079                                  <1>       ;    p.dska - holds block number of process
13080                                  <1>       ; OUTPUTS ->
13081                                  <1>       ;    swp I/O queue
13082                                  <1>       ;    p.break - negative word count of process
13083                                  <1>       ;    r1 - process disk address
13084                                  <1>       ;    r2 - negative word count
13085                                  <1>       ;
13086                                  <1>       ; RETRO UNIX 8086 v1 input/output:
13087                                  <1>       ;
13088                                  <1>       ; INPUTS ->
13089                                  <1>       ;    u.uno - process number (to be swapped out)
13090                                  <1>       ; OUTPUTS ->
13091                                  <1>       ;    none
13092                                  <1>       ;
13093                                  <1>       ;   ((Modified registers: ECX, ESI, EDI))
13094                                  <1>       ;
13095                                  <1>
13096                                  <1>       ; 28/02/2017
13097                                  <1>       ;cmp   byte [multi_tasking], 0  ; Musti tasking mode ?
13098                                  <1>       ;jna   short wswp
13099 00011178 803D[DA030300]00       <1>       cmp    byte [u.fpsave], 0 ; 28/02/2017
13100 0001117F 7606                   <1>       jna    short wswp
13101 00011181 DD35[DC030300]         <1>       fnsave [u.fpregs] ; save floating point registers (94 bytes)
13102                                  <1> wswp:
13103 00011187 8B3D[B4030300]         <1>       mov    edi, [u.upage] ; process's user (u) structure page addr
13104 0001118D B938000000             <1>       mov    ecx, (U_SIZE + 3) / 4
13105 00011192 BE[5C030300]           <1>       mov    esi, user ; active user (u) structure
13106 00011197 F3A5                   <1>       rep    movsd
13107                                  <1>       ;
13108 00011199 8B35[60030300]         <1>       mov    esi, [u.usp] ; esp (system stack pointer,
13109                                  <1>                           ;      points to user registers)
13110 0001119F 8B0D[5C030300]         <1>       mov    ecx, [u.sp]  ; return address from the system call
13111                                  <1>                           ; (for IRET)
13112                                  <1>                           ; [u.sp] -> EIP (user)
13113                                  <1>                           ; [u.sp+4]-> CS (user)
13114                                  <1>                           ; [u.sp+8] -> EFLAGS (user)
13115                                  <1>                           ; [u.sp+12] -> ESP (user)
13116                                  <1>                           ; [u.sp+16] -> SS (user)
13117 000111A5 29F1                   <1>       sub    ecx, esi     ; required space for user registers
13118 000111A7 83C114                 <1>       add    ecx, 20           ; +5 dwords to return from system call
13119                                  <1>                           ; (for IRET)
13120 000111AA C1E902                 <1>       shr    ecx, 2
13121 000111AD F3A5                   <1>       rep    movsd
13122 000111AF C3                     <1>       retn
13123                                  <1>
13124                                  <1> rswap:  ; < swap in, swap from disk >
13125                                  <1>       ; 28/02/2017 (frstor)
13126                                  <1>       ; 15/01/2017
13127                                  <1>       ; 14/01/2017
13128                                  <1>       ; 21/05/2016
13129                                  <1>       ; 03/05/2016
13130                                  <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
13131                                  <1>       ; 09/05/2015 - 15/09/2015 (Retro UNIX 386 v1)
13132                                  <1>       ; 26/05/2013 - 08/03/2014 (Retro UNIX 8086 v1)
13133                                  <1>       ; 'rswap' reads a process whose number is in r1,
13134                                  <1>       ; from disk into core.
13135                                  <1>       ;
13136                                  <1>       ; Retro UNIX 386 v1 modification ->
13137                                  <1>       ;       User (u) structure content and the user's register content
13138                                  <1>       ;     will be restored from process's/user's UPAGE (a page for
13139                                  <1>       ;     saving 'u' structure and user registers for task switching).
13140                                  <1>       ;     u.usp - points to kernel stack address which contains
13141                                  <1>       ;             user's registers while entering system call.
13142                                  <1>       ;     u.sp  - points to kernel stack address
13143                                  <1>       ;             to return from system call -for IRET-.
13144                                  <1>       ;     [u.usp]+32+16 = [u.sp]
13145                                  <1>       ;     [u.usp] -> edi, esi, ebp, esp (= [u.usp]+32), ebx,
13146                                  <1>       ;             edx, ecx, eax, gs, fs, es, ds, -> [u.sp].
13147                                  <1>       ;
13148                                  <1>       ; RETRO UNIX 8086 v1 modification ->
13149                                  <1>       ;       'swap to disk' is replaced with 'change running segment'
13150                                  <1>       ;     according to 8086 cpu (x86 real mode) architecture.
13151                                  <1>       ;     pdp-11 was using 64KB uniform memory while IBM PC
13152                                  <1>       ;     compatibles was using 1MB segmented memory
13153                                  <1>       ;     in 8086/8088 times.
13154                                  <1>       ;
13155                                  <1>       ; INPUTS ->
13156                                  <1>       ;    r1 - process number of process to be read in
13157                                  <1>       ;    p.break - negative of word count of process
13158                                  <1>       ;    p.dska - disk address of the process
13159                                  <1>       ;    u.emt - determines handling of emt's
13160                                  <1>       ;    u.ilgins - determines handling of illegal instructions
13161                                  <1>       ; OUTPUTS ->
13162                                  <1>       ;    8 = (u.ilgins)
13163                                  <1>       ;    24 = (u.emt)
13164                                  <1>       ;    swp - bit 10 is set to indicate read
13165                                  <1>       ;             (bit 15=0 when reading is done)
13166                                  <1>       ;    swp+2 - disk block address
13167                                  <1>       ;    swp+4 - negative word count
13168                                  <1>       ;      ((swp+6 - address of user structure))
13169                                  <1>       ;
13170                                  <1>       ; RETRO UNIX 8086 v1 input/output:
13171                                  <1>       ;
13172                                  <1>       ; INPUTS ->
13173                                  <1>       ;    AL      - new process number (to be swapped in)
13174                                  <1>       ; OUTPUTS ->
13175                                  <1>       ;    none
13176                                  <1>       ;
```

```
13177                                  <1>       ;   ((Modified registers: EAX, ECX, ESI, EDI, ESP))
13178                                  <1>       ;
13179                                  <1>       ; Retro UNIX 386 v1 - modification ! 14/05/2015
13180 000111B0 89C6                    <1>       mov   esi, eax  ; process's user (u) structure page addr
13181 000111B2 B938000000              <1>       mov   ecx, (U_SIZE + 3) / 4
13182 000111B7 BF[5C030300]            <1>       mov   edi, user ; active user (u) structure
13183 000111BC F3A5                    <1>       rep   movsd
13184 000111BE 58                      <1>       pop   eax    ; 'rswap' return address
13185                                  <1>       ;
13186                                  <1>       ;cli
13187 000111BF 8B3D[60030300]          <1>       mov   edi, [u.usp] ; esp (system stack pointer,
13188                                  <1>                          ;     points to user registers)
13189 000111C5 89FC                    <1>       mov   esp, edi     ; 14/01/2017
13190 000111C7 8B0D[5C030300]          <1>       mov   ecx, [u.sp]  ; return address from the system call
13191                                  <1>                          ; (for IRET)
13192                                  <1>                          ; [u.sp] -> EIP (user)
13193                                  <1>                          ; [u.sp+4]-> CS (user)
13194                                  <1>                          ; [u.sp+8] -> EFLAGS (user)
13195                                  <1>                          ; [u.sp+12] -> ESP (user)
13196                                  <1>                          ; [u.sp+16] -> SS (user)
13197 000111CD 29F9                    <1>       sub   ecx, edi     ; required space for user registers
13198 000111CF 83C114                  <1>       add   ecx, 20           ; +5 dwords to return from system call
13199                                  <1>                          ; (for IRET)
13200 000111D2 C1E902                  <1>       shr   ecx, 2
13201 000111D5 F3A5                    <1>       rep   movsd
13202                                  <1>       ;mov  esp, [u.usp] ; 15/09/2015
13203                                  <1>       ;sti
13204                                  <1>       ; 28/02/2017
13205                                  <1>       ;cmp  byte [multi_tasking], 0  ; Musti tasking mode ?
13206                                  <1>       ;jna  short rswp_retn
13207 000111D7 803D[DA030300]00        <1>       cmp   byte [u.fpsave], 0
13208 000111DE 7606                    <1>       jna   short rswp_retn
13209 000111E0 DD25[DC030300]          <1>       frstor [u.fpregs] ; restore floating point regs (94 bytes)
13210                                  <1> rswp_retn:
13211 000111E6 50                      <1>       push  eax    ; 'rswap' return address
13212 000111E7 C3                      <1>       retn
13213                                  <1>
13214                                  <1> putlu:
13215                                  <1>       ; 20/05/2016
13216                                  <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
13217                                  <1>       ; 10/05/2015 - 12/09/2015 (Retro UNIX 386 v1)
13218                                  <1>       ; 15/04/2013 - 23/02/2014 (Retro UNIX 8086 v1)
13219                                  <1>       ; 'putlu' is called with a process number in r1 and a pointer
13220                                  <1>       ; to lowest priority Q (runq+4) in r2. A link is created from
13221                                  <1>       ; the last process on the queue to process in r1 by putting
13222                                  <1>       ; the process number in r1 into the last process's link.
13223                                  <1>       ;
13224                                  <1>       ; INPUTS ->
13225                                  <1>       ;    r1 - user process number
13226                                  <1>       ;    r2 - points to lowest priority queue
13227                                  <1>       ;    p.dska - disk address of the process
13228                                  <1>       ;    u.emt - determines handling of emt's
13229                                  <1>       ;    u.ilgins - determines handling of illegal instructions
13230                                  <1>       ; OUTPUTS ->
13231                                  <1>       ;    r3 - process number of last process on the queue upon
13232                                  <1>       ;       entering putlu
13233                                  <1>       ;    p.link-1 + r3 - process number in r1
13234                                  <1>       ;    r2 - points to lowest priority queue
13235                                  <1>       ;
13236                                  <1>       ; ((Modified registers: EDX, EBX))
13237                                  <1>       ;
13238                                  <1>       ; / r1 = user process no.; r2 points to lowest priority queue
13239                                  <1>
13240                                  <1>       ; EBX = r2
13241                                  <1>       ; EAX = r1 (AL=r1b)
13242                                  <1>
13243                                  <1>       ; 20/05/2016
13244                                  <1>       ; AL = process number (1 to 16) // Retro UNIX 8086, 386 v1 //
13245                                  <1>       ;     (max. 16 processes available for current kernel version)
13246                                  <1>       ; EBX = run queue address ; 20/05/2016 (TRDOS 386)
13247                                  <1>            ; which is one of following addresses:
13248                                  <1>            ;  1) 'runq_event' high priority run queue
13249                                  <1>            ;  2) 'runq_normal' normal/regular priority run queue
13250                                  <1>            ;  3) 'runq_background' low priority run queue
13251                                  <1>
13252                                  <1>       ;mov  ebx, runq
13253 000111E8 0FB613                  <1>       movzx       edx, byte [ebx]
13254 000111EB 43                      <1>       inc   ebx
13255 000111EC 20D2                    <1>       and   dl, dl
13256                                  <1>            ; tstb (r2)+ / is queue empty?
13257 000111EE 740A                    <1>            jz    short putlu_1
13258                                  <1>            ; beq 1f / yes, branch
13259 000111F0 8A13                    <1>       mov   dl, [ebx] ; 12/09/2015
13260                                  <1>            ; movb (r2),r3 / no, save the "last user" process number
13261                                  <1>                       ; / in r3
13262 000111F2 8882[9F000300]          <1>            mov   [edx+p.link-1], al
13263                                  <1>            ; movb r1,p.link-1(r3) / put pointer to user on
13264                                  <1>                       ; / "last users" link
13265 000111F8 EB03                    <1>       jmp   short putlu_2
13266                                  <1>            ; br 2f /
13267                                  <1> putlu_1: ; 1:
13268 000111FA 8843FF                  <1>       mov   [ebx-1], al
13269                                  <1>                  ; movb r1,-1(r2) / user is only user;
13270                                  <1>                      ; / put process no. at beginning and at end
13271                                  <1> putlu_2: ; 2:
13272 000111FD 8803                    <1>       mov   [ebx], al
13273                                  <1>                  ; movb r1,(r2) / user process in r1 is now the last entry
13274                                  <1>                       ; / on the queue
13275 000111FF 88C2                    <1>       mov   dl, al
13276 00011201 88B2[9F000300]          <1>         mov     [edx+p.link-1], dh ; 0
13277                                  <1>            ; dec r2 / restore r2
13278 00011207 C3                      <1>         retn
13279                                  <1>            ; rts r0
13280                                  <1>
13281                                  <1> sysver:
```

```
13282                                 <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
13283 00011208 C705[64030300]0002-    <1>       mov    dword [u.r0], 200h ; AH = major version, AL = minor version
13283 00011210 0000                   <1>
13284 00011212 E9D5C6FFFF             <1>       jmp    sysret
13285                                 <1>
13286                                 <1>
13287                                 <1> syspri: ; change running priority (of the process)
13288                                 <1>       ; 21/05/2016
13289                                 <1>       ; 20/05/2026 - TRDOS 386 (TRDOS v2.0)
13290                                 <1>       ; INPUT ->
13291                                 <1>       ;     BL = priority level
13292                                 <1>       ;        0 = low running priority (running on background)
13293                                 <1>       ;        1 = normal/regular priority (running as regular)
13294                                 <1>       ;        2 = high/event priority (running for event)
13295                                 <1>       ;        >2 = invalid, it will accepted as 2 (event)
13296                                 <1>       ;        0FFh = get/return current running priority only
13297                                 <1>       ; OUTPUT ->
13298                                 <1>       ;     * if current [u.pri] < 2
13299                                 <1>       ;       if BL input < 0FFh ->
13300                                 <1>       ;          [u.pri] is updated as in BL input (0,1,2)
13301                                 <1>       ;       if BL input = 0FFh -> AL = [u.pri] (current)
13302                                 <1>       ;
13303                                 <1>       ;     * if current [u.pri] = 2
13304                                 <1>       ;       if BL input < 0FFh -> cf = 1 & AL = 2
13305                                 <1>       ;       if BL input = 0FFh -> cf = 0 & AL = 2
13306                                 <1>       ;
13307                                 <1>       ;     NOTE:
13308                                 <1>       ;     If [u.pri] = 2, it can not be changed to 1 or 0;
13309                                 <1>       ;     because, run queue of the running process is unspecified
13310                                 <1>       ;     at this      stage. Process might be started by a timer event
13311                                 <1>       ;     or priority might be changed to high by previous
13312                                 <1>       ;     'syspri' system     call. In both cases, the process is in
13313                                 <1>       ;     'runq_normal' or 'runq_background' queue.
13314                                 <1>       ;     As result of this fact, when the [u.quant] time quantum
13315                                 <1>       ;     of the process is elapsed or 'sysrele' system call is
13316                                 <1>       ;     instructed by the process, 'tswap' ('tswitch') procedure
13317                                 <1>       ;     will be called (to 'swap' or 'switch' out the procedure)
13318                                 <1>       ;     and it will not call 'putlu' to add the (stopping)
13319                                 <1>       ;     process to relevant run queue when [u.pri] = 2.
13320                                 <1>       ;     (Otherwise, it would be possible to add process to
13321                                 <1>       ;     a run queue while it is already in a run queue, wrongly.)
13322                                 <1>       ;
13323                                 <1>       ;     If [u.pri]< 2, 'tswap/tswitch' procedure will call
13324                                 <1>       ;     'putlu' to add process to relevant run queue
13325                                 <1>       ;     according to [u.pri] value. ('runq_normal' for 1,
13326                                 <1>       ;     'runq_background' for 0).
13327                                 <1>       ;
13328                                 <1>       ;     If BL input >= 2 and < 0FFh while [u.pri] < 2,
13329                                 <1>       ;     process will be added to 'runq_normal' queue and
13330                                 <1>       ;     [u.pri] will be set to 2. (in 'syspri' system call)
13331                                 <1>       ;
13332                                 <1>
13333 00011217 29C0                  <1>       sub    eax, eax ; 0
13334 00011219 A3[C8030300]          <1>       mov    [u.error], eax
13335                                 <1>
13336 0001121E A0[A9030300]          <1>       mov    al, [u.pri]
13337 00011223 A3[64030300]          <1>       mov    [u.r0], eax
13338                                 <1>
13339 00011228 FEC3                  <1>       inc    bl
13340 0001122A 0F84BCC6FFFF          <1>       jz     sysret ; 0FFh -> 0, get priority level
13341                                 <1>
13342 00011230 3C02                  <1>       cmp    al, 2
13343 00011232 0F8394C6FFFF          <1>       jnb    error ; CF = 1 & AL = 2 (& last error = 0)
13344                                 <1>
13345 00011238 FECB                  <1>       dec    bl
13346 0001123A 80FB02                <1>       cmp    bl, 2
13347 0001123D 7602                  <1>       jna    short syspri_1
13348 0001123F B302                  <1>       mov    bl, 2
13349                                 <1> syspri_1:
13350 00011241 881D[A9030300]        <1>       mov    [u.pri], bl
13351 00011247 80FB02                <1>       cmp    bl, 2
13352 0001124A 0F829CC6FFFF          <1>         jb     sysret
13353                                 <1>
13354                                 <1>       ; here...
13355                                 <1>       ; Priority of current process has been changed to high
13356                                 <1>       ; ('run for event') but current process will be added to
13357                                 <1>       ; 'run as normal' queue. ('run for event' high priority
13358                                 <1>       ; queue is under control of timer -& RTC- interrupt only!)
13359                                 <1>       ;
13360                                 <1>       ; (Otherwise, process can fall into black hole!
13361                                 <1>       ; e.g. if it is not in waiting list and it has not got
13362                                 <1>       ; a timer event and it is not in a run queue!
13363                                 <1>       ; Because, when [u.pri] is 2, 'tswap/tswitch' will not
13364                                 <1>       ; add the stopping process to a run queue.)
13365                                 <1>
13366 00011250 A0[B3030300]          <1>       mov    al, [u.uno]
13367 00011255 BB[54030300]          <1>       mov    ebx, runq_normal ; normal priority !
13368                                 <1>                             ; [u.pri] is set to high
13369                                 <1>                             ; but 'runq_event' queue is set
13370                                 <1>                             ; only by the kernel's timer
13371                                 <1>                             ; event function (timer interrupt).
13372 0001125A E889FFFFFF            <1>       call   putlu
13373 0001125F E988C6FFFF            <1>       jmp    sysret
13374                                 <1>
13375                                 <1> cpass: ; / get next character from user area of core and put it in AL (r1)
13376                                 <1>       ; 02/05/2016 - TRDOS 386 (TRDOS v2.0)
13377                                 <1>       ; 19/05/2015 - 18/10/2015 (Retro UNIX 386 v1)
13378                                 <1>       ; 14/08/2013 - 20/09/2013 (Retro UNIX 8086 v1)
13379                                 <1>       ; INPUTS ->
13380                                 <1>       ;     [u.base] = virtual address in user area
13381                                 <1>       ;     [u.count] = byte count (max.)
13382                                 <1>       ;     [u.pcount] = byte count in page (0 = reset)
13383                                 <1>       ; OUTPUTS ->
13384                                 <1>       ;     AL = the character which is pointed by [u.base]
13385                                 <1>       ;     zf = 1 -> transfer count has been completed
```

```
13386                                  <1>         ;
13387                                  <1>       ; ((Modified registers:  EAX, EDX, ECX))
13388                                  <1>       ;
13389 00011264 833D[88030300]00        <1>       cmp    dword [u.count], 0  ; have all the characters been transferred
13390                                  <1>                                  ; i.e., u.count, # of chars. left
13391 0001126B 763F                    <1>       jna    short cpass_3    ; to be transferred = 0?) yes, branch
13392 0001126D FF0D[88030300]          <1>       dec    dword [u.count]         ; no, decrement u.count
13393                                  <1>         ; 19/05/2015
13394                                  <1>       ;(Retro UNIX 386 v1 - translation from user's virtual address
13395                                  <1>       ;                  to physical address
13396 00011273 66833D[C4030300]00      <1>       cmp    word [u.pcount], 0 ; byte count in page = 0 (initial value)
13397                                  <1>                           ; 1-4095 --> use previous physical base address
13398                                  <1>                           ; in [u.pbase]
13399 0001127B 770E                    <1>       ja     short cpass_1
13400 0001127D 833D[BC030300]00        <1>       cmp     dword [u.ppgdir], 0 ; is the caller os kernel
13401 00011284 7427                    <1>         je      short cpass_k       ; (sysexec, '/etc/init') ?  (MainProg)
13402 00011286 E858FDFFFF              <1>       call   trans_addr_r
13403                                  <1> cpass_1:
13404 0001128B 66FF0D[C4030300]        <1>       dec    word [u.pcount]
13405                                  <1> cpass_2:
13406 00011292 8B15[C0030300]          <1>       mov    edx, [u.pbase]
13407 00011298 8A02                    <1>       mov    al, [edx]    ; take the character pointed to
13408                                  <1>                           ; by u.base and put it in r1
13409 0001129A FF05[8C030300]          <1>       inc    dword [u.nread] ; increment no. of bytes transferred
13410 000112A0 FF05[84030300]          <1>       inc    dword [u.base]  ; increment the buffer address to point to the
13411                                  <1>                             ; next byte
13412 000112A6 FF05[C0030300]          <1>       inc    dword [u.pbase]
13413                                  <1> cpass_3:
13414 000112AC C3                      <1>       retn
13415                                  <1> cpass_k:
13416                                  <1>       ; 02/07/2015
13417                                  <1>       ; The caller is os kernel
13418                                  <1>       ; (get sysexec arguments from kernel's memory space)
13419 000112AD 8B1D[84030300]          <1>       mov    ebx, [u.base]
13420 000112B3 66C705[C4030300]00-     <1>         mov     word [u.pcount], PAGE_SIZE ; 4096
13420 000112BB 10                      <1>
13421 000112BC 891D[C0030300]          <1>       mov    [u.pbase], ebx
13422 000112C2 EBCE                    <1>       jmp    short cpass_2
13423                                  <1>
13424                                  <1> transfer_to_user_buffer: ; fast transfer
13425                                  <1>       ; 27/05/2016
13426                                  <1>       ; 16/05/2016 - TRDOS 386 (TRDOS v2.0)
13427                                  <1>       ;
13428                                  <1>       ; INPUT ->
13429                                  <1>       ;      ESI = source address in system space
13430                                  <1>       ;      EDI = user's buffer address
13431                                  <1>       ;      ECX = transfer (byte) count
13432                                  <1>       ;      [u.pgdir] = user's page directory
13433                                  <1>       ; OUTPUT ->
13434                                  <1>       ;      ECX = actual transfer count
13435                                  <1>       ;      cf = 1 -> error
13436                                  <1>       ;      [u.count] = remain byte count
13437                                  <1>       ;
13438                                  <1>       ; Modified registers: eax, ecx
13439                                  <1>       ;
13440                                  <1>
13441 000112C4 21C9                    <1>       and    ecx, ecx
13442 000112C6 743B                    <1>       jz     short ttub_4
13443                                  <1>
13444 000112C8 890D[88030300]          <1>       mov    [u.count], ecx
13445                                  <1>
13446 000112CE 57                      <1>       push   edi
13447 000112CF 56                      <1>       push   esi
13448 000112D0 53                      <1>       push   ebx
13449 000112D1 52                      <1>       push   edx
13450 000112D2 51                      <1>       push   ecx
13451                                  <1>
13452 000112D3 89FB                    <1>       mov    ebx, edi
13453 000112D5 81C300004000            <1>       add    ebx, CORE ; 27/05/2016
13454                                  <1> ttub_1:
13455                                  <1>       ; ebx = virtual (linear) address
13456                                  <1>       ; [u.pgdir] = user's page directory
13457 000112DB E80450FFFF              <1>         call   get_physical_addr_x ; get physical address
13458 000112E0 7222                    <1>       jc     short ttub_5
13459                                  <1>       ; eax = physical address
13460                                  <1>       ; ecx = remain byte count in page (1-4096)
13461 000112E2 89C7                    <1>       mov    edi, eax
13462 000112E4 A1[88030300]            <1>       mov    eax, [u.count]
13463 000112E9 39C1                    <1>       cmp    ecx, eax
13464 000112EB 7602                    <1>       jna    short ttub_2
13465 000112ED 89C1                    <1>       mov    ecx, eax
13466                                  <1> ttub_2:
13467 000112EF 29C8                    <1>       sub    eax, ecx
13468 000112F1 01CB                    <1>       add    ebx, ecx
13469 000112F3 F3A4                    <1>       rep    movsb
13470 000112F5 A3[88030300]            <1>       mov    [u.count], eax
13471 000112FA 09C0                    <1>       or     eax, eax
13472 000112FC 75DD                    <1>       jnz    short ttub_1
13473                                  <1> ttub_retn:
13474                                  <1> tfub_retn:
13475 000112FE 59                      <1>       pop    ecx ; transfer count = actual transfer count
13476                                  <1> ttub_3:
13477 000112FF 5A                      <1>       pop    edx
13478 00011300 5B                      <1>       pop    ebx
13479 00011301 5E                      <1>       pop    esi
13480 00011302 5F                      <1>       pop    edi
13481                                  <1> ttub_4:
13482 00011303 C3                      <1>       retn
13483                                  <1> ttub_5:
13484 00011304 59                      <1>       pop    ecx
13485 00011305 2B0D[88030300]          <1>       sub    ecx, [u.count] ; actual transfer count
13486 0001130B F9                      <1>       stc
13487 0001130C EBF1                    <1>       jmp    short ttub_3
13488                                  <1>
13489                                  <1> transfer_from_user_buffer: ; fast transfer
```

```
13490                                  <1>       ; 27/05/2016
13491                                  <1>       ; 16/05/2016 - TRDOS 386 (TRDOS v2.0)
13492                                  <1>       ;
13493                                  <1>       ; INPUT ->
13494                                  <1>       ;      ESI = user's buffer address
13495                                  <1>       ;      EDI = destination address in system space
13496                                  <1>       ;      ECX = transfer (byte) count
13497                                  <1>       ;      [u.pgdir] = user's page directory
13498                                  <1>       ; OUTPUT ->
13499                                  <1>       ;      ecx = actual transfer count
13500                                  <1>       ;      cf = 1 -> error
13501                                  <1>       ;      [u.count] = remain byte count
13502                                  <1>       ;
13503                                  <1>       ; Modified registers: eax, ecx
13504                                  <1>       ;
13505                                  <1>
13506 0001130E 21C9                   <1>       and    ecx, ecx
13507                                  <1>       ;jz    short tfub_4
13508 00011310 74F1                   <1>       jz     short ttub_4
13509                                  <1>
13510 00011312 890D[88030300]         <1>       mov    [u.count], ecx
13511                                  <1>
13512 00011318 57                     <1>       push   edi
13513 00011319 56                     <1>       push   esi
13514 0001131A 53                     <1>       push   ebx
13515 0001131B 52                     <1>       push   edx
13516 0001131C 51                     <1>       push   ecx
13517                                  <1>
13518 0001131D 89F3                   <1>       mov    ebx, esi
13519 0001131F 81C300004000           <1>       add    ebx, CORE ; 27/05/2016
13520                                  <1> tfub_1:
13521                                  <1>       ; ebx = virtual (linear) address
13522                                  <1>       ; [u.pgdir] = user's page directory
13523 00011325 E8BA4FFFFF             <1>             call   get_physical_addr_x ; get physical address
13524                                  <1>       ;jc    short tfub_5
13525 0001132A 72D8                   <1>       jc     short ttub_5
13526                                  <1>       ; eax = physical address
13527                                  <1>       ; ecx = remain byte count in page (1-4096)
13528 0001132C 89C6                   <1>       mov    esi, eax
13529 0001132E A1[88030300]           <1>       mov    eax, [u.count]
13530 00011333 39C1                   <1>       cmp    ecx, eax
13531 00011335 7602                   <1>       jna    short tfub_2
13532 00011337 89C1                   <1>       mov    ecx, eax
13533                                  <1> tfub_2:
13534 00011339 29C8                   <1>       sub    eax, ecx
13535 0001133B 01CB                   <1>       add    ebx, ecx
13536 0001133D F3A4                   <1>       rep    movsb
13537 0001133F A3[88030300]           <1>       mov    [u.count], eax
13538 00011344 09C0                   <1>       or     eax, eax
13539 00011346 75DD                   <1>       jnz    short tfub_1
13540                                  <1>
13541 00011348 EBB4                   <1>       jmp    short tfub_retn
13542                                  <1>
13543                                  <1> ;tfub_retn:
13544                                  <1> ;     pop    ecx ; transfer count = actual transfer count
13545                                  <1> ;tfub_3:
13546                                  <1> ;     pop    edx
13547                                  <1> ;     pop    ebx
13548                                  <1> ;     pop    esi
13549                                  <1> ;     pop    edi
13550                                  <1> ;tfub_4:
13551                                  <1> ;     retn
13552                                  <1> ;tfub_5:
13553                                  <1> ;     pop    ecx
13554                                  <1> ;     sub    ecx, [u.count] ; actual transfer count
13555                                  <1> ;     stc
13556                                  <1> ;     jmp    short tfub_3
13557                                  <1>
13558                                  <1> sysfff: ; <Find First File>
13559                                  <1>       ; 17/10/2016
13560                                  <1>       ; 16/10/2016
13561                                  <1>       ; 15/10/2016 TRDOS 386 (TRDOS v2.0) feature only !
13562                                  <1>       ;           -derived from TRDOS v1.0, INT_21H.ASM-
13563                                  <1>       ;            ("loc_INT21h_find_first_file")
13564                                  <1>       ; TRDOS 8086 (v1.0)
13565                                  <1>         ;   07/08/2011
13566                                  <1>         ;   Find First File
13567                                  <1>       ;     INPUT:
13568                                  <1>         ;       CX= Attributes
13569                                  <1>         ;       DS:DX= Pointer to filename
13570                                  <1>       ;     MSDOS OUTPUT:
13571                                  <1>       ;         DTA: (Default address: PSP offset 80h)
13572                                  <1>       ;         Offset  Descrription
13573                                  <1>       ;         0      Reserved for use find next file
13574                                  <1>       ;         21     Attribute of file found
13575                                  <1>       ;         22     Time stamp of file
13576                                  <1>       ;         24     Date stamp of file
13577                                  <1>       ;         26     File size in bytes
13578                                  <1>       ;         30     Filename and extension (zero terminated)
13579                                  <1>       ;     If cf = 1:
13580                                  <1>       ;         Error Codes: (in AX)
13581                                  <1>       ;            2 - File not found
13582                                  <1>       ;            18 - No more files
13583                                  <1>         ;
13584                                  <1>       ; TRDOS 386 (v2.0)
13585                                  <1>       ; 15/10/2016
13586                                  <1>       ;
13587                                  <1>         ; INPUT ->
13588                                  <1>         ;      CL = File attributes
13589                                  <1>       ;           bit 0 (1) - Read only file (R)
13590                                  <1>       ;            bit 1 (1) - Hidden file (H)
13591                                  <1>         ;            bit 2 (1) - System file (R)
13592                                  <1>       ;            bit 3 (1) - Volume label/name (V)
13593                                  <1>         ;            bit 4 (1) - Subdirectory (D)
13594                                  <1>       ;           bit 5 (1) - File has been archived (A)
```

```
13595                                  <1>       ;         CH = 0 -> Return basic parameters (24 bytes)
13596                                  <1>       ;         CH > 0 -> Return FindFile structure/table (128 bytes)
13597                                  <1>         ;           EBX = Pointer to filename (ASCIIZ) -path-
13598                                  <1>       ;         EDX = File parameters buffer address
13599                                  <1>       ;             (buffer size = 24 bytes if CH input = 0)
13600                                  <1>       ;             (buffer size = 128 bytes if CH input > 0)
13601                                  <1>       ;
13602                                  <1>       ; OUTPUT ->
13603                                  <1>       ;         EAX = 0 if CH input > 0
13604                                  <1>       ;         EAX = First cluster number of file if CH input = 0
13605                                  <1>       ;         EDX = File parameters table/structure address
13606                                  <1>       ;         Basic Parameters:
13607                                  <1>       ;             Offset  Description
13608                                  <1>       ;             ------ ---------------
13609                                  <1>       ;             0      File Attributes
13610                                  <1>       ;             1      Ambiguous filename chars are used sign
13611                                  <1>       ;                    (0 = filename fits exactly with request)
13612                                  <1>       ;                    (>0 = ambiguous filename chars are used)
13613                                  <1>       ;             2      Time stamp of file
13614                                  <1>       ;             4      Date stamp of file
13615                                  <1>       ;             6      File size in bytes
13616                                  <1>       ;             10     Short Filename (ASCIIZ, max. 13 bytes)
13617                                  <1>       ;             23     Longname Length (1-255) if existing
13618                                  <1>       ;
13619                                  <1>       ;           cf = 1 -> Error code in AL
13620                                  <1>       ;
13621                                  <1>       ; Modified Registers: EAX (at the return of system call)
13622                                  <1>       ;
13623                                  <1>       ; TR-DOS FindFile (FFF) Structure (128 bytes):
13624                                  <1>       ; 09/10/2011 (DIR.ASM) - 10/02/2016 (trdoskx.s)
13625                                  <1>       ;
13626                                  <1>       ; Offset     Parameter           Size
13627                                  <1>       ; ------     ------------------  --------
13628                                  <1>       ; 0          FindFile_Drv        1 byte
13629                                  <1>       ; 1          FindFile_Directory  65 bytes
13630                                  <1>       ; 66         FindFile_Name       13 bytes
13631                                  <1>       ; 79         FindFile_LongNameEntryLength 1 byte
13632                                  <1>       ;Above 80 bytes form
13633                                  <1>       ;TR-DOS Source/Destination File FullName Format/Structure
13634                                  <1>       ; 80         FindFile_AttributesMask 1 word
13635                                  <1>       ; 82         FindFile_DirEntry   32 bytes (*)
13636                                  <1>       ; 114        FindFile_DirFirstCluster 1 double word
13637                                  <1>       ; 118        FindFile_DirCluster 1 double word
13638                                  <1>       ; 122        FindFile_DirEntryNumber 1 word
13639                                  <1>       ; 124        FindFile_MatchCounter     1 word
13640                                  <1>       ; 126        FindFile_Reserved   1 word
13641                                  <1>       ; (*) MS-DOS, FAT 12-16-32 classic directory entry (32 bytes)
13642                                  <1>
13643                                  <1>       ;mov   [u.namep], ebx
13644                                  <1>       ; 16/10/2016
13645 0001134A 8915[408E0100]         <1>       mov    [FFF_UBuffer], edx
13646 00011350 66890D[458E0100]       <1>       mov    [FFF_Attrib], cx ; [FFF_RType] = ch
13647                                  <1>                   ; Attributes in CL, return data type in CH
13648 00011357 89DE                   <1>       mov    esi, ebx
13649                                  <1>       ; file name is forced, change directory as temporary
13650                                  <1>       ;mov   ax, 1
13651                                  <1>       ;mov   [FFF_Valid], ah ; 0 ; reset ; 17/10/2016
13652                                  <1>       ;call  set_working_path
13653 00011359 E8E2130000             <1>       call   set_working_path_x ; 17/10/2016
13654 0001135E 731D                   <1>       jnc    short sysfff_0
13655                                  <1>
13656 00011360 21C0                   <1>       and    eax, eax  ; 0 -> Bad Path!
13657 00011362 7505                   <1>       jnz    short sysfff_err
13658                                  <1>
13659                                  <1>       ; eax = 0
13660 00011364 B80C000000             <1>       mov    eax, ERR_DIR_NOT_FOUND ; Directory not found !
13661                                  <1> sysfff_err:
13662 00011369 A3[64030300]           <1>       mov    [u.r0], eax
13663 0001136E A3[C8030300]           <1>       mov    [u.error], eax
13664 00011373 E89D140000             <1>         call       reset_working_path
13665 00011378 E94FC5FFFF             <1>       jmp    error
13666                                  <1>
13667                                  <1> sysfff_0:
13668                                  <1>       ;sub   ah, ah ; ah = 0
13669 0001137D 8A0424                 <1>       mov    al, [esp]
13670 00011380 08C0                   <1>       or     al, al
13671 00011382 7412                   <1>       jz     short sysfff_2
13672 00011384 B410                   <1>       mov    ah, 10h
13673 00011386 A808                   <1>       test   al, 08h
13674 00011388 7503                   <1>       jnz    short sysfff_1
13675 0001138A 80CC08                 <1>       or     ah, 08h
13676                                  <1> sysfff_1:
13677 0001138D 2410                   <1>       and    al, 10h ; Directory
13678 0001138F 7405                   <1>       jz     short sysfff_2
13679 00011391 80E408                 <1>       and    ah, 08h
13680 00011394 30C0                   <1>       xor    al, al ; When a directory is searched,
13681                                  <1>                   ; filename will be returned even if
13682                                  <1>                   ; it is not a directory!
13683                                  <1>                   ; Because: (in order to prevent
13684                                  <1>                   ; creating a dir with existing file name)
13685                                  <1>                   ; Dir and file names must not be same!
13686                                  <1>                   ; (return attribute must be checked)
13687                                  <1> sysfff_2:
13688                                  <1>       ; AX = Attributes mask
13689                                  <1>             ; AL = AND mask (result must be equal to AL)
13690                                  <1>             ; AH = Negative AND mask (result must be ZERO)
13691                                  <1>       ; ESI = FindFile_Name address
13692                                  <1>
13693 00011396 E8D680FFFF             <1>       call   find_first_file
13694 0001139B 72CC                   <1>       jc     short sysfff_err ; eax = 2 (File not found !)
13695                                  <1>
13696                                  <1>       ; ESI = Directory Entry (FindFile_DirEntry) Location
13697                                  <1>       ; EDI = Directory Buffer Directory Entry Location
13698                                  <1>       ; EAX = File Size
13699                                  <1>       ;  BL = Attributes of The File/Directory
```

```
13700                                  <1>       ;  BH = Long Name Yes/No Status (>0 is YES)
13701                                  <1>       ;  DX > 0 : Ambiguous filename chars are used
13702                                  <1>
13703                                  <1> sysfff_3:
13704                                  <1>       ; 16/10/2016
13705 0001139D 668B0D[458E0100]        <1>       mov    cx, [FFF_Attrib]
13706                                  <1>       ; Attribs in CL, return data type in CH
13707                                  <1>
13708                                  <1>       ;or    cl, cl
13709                                  <1>       ;jz    short sysfff_4 ; 0 = No filter
13710 000113A4 80F1FF                 <1>       xor    cl, 0FFh
13711 000113A7 20D9                   <1>       and    cl, bl
13712 000113A9 7409                   <1>       jz     short sysfff_4
13713                                  <1>
13714                                  <1>       ;mov   eax, 2 ; 'file not found !' error
13715                                  <1>       ;jmp   short sysfff_err_1
13716                                  <1>
13717                                  <1>       ; 16/10/2016
13718 000113AB E87081FFFF             <1>       call   find_next_file
13719 000113B0 72B7                   <1>       jc     short sysfff_err ; eax = 12 (no more files !)
13720 000113B2 EBE9                   <1>       jmp    short sysfff_3
13721                                  <1>
13722                                  <1> sysfff_4:
13723 000113B4 20ED                   <1>       and    ch, ch ; [FFF_RType]
13724 000113B6 7412                   <1>       jz     short sysfff_5
13725 000113B8 B980000000             <1>       mov    ecx, 128 ; ; transfer length
13726 000113BD 880D[448E0100]         <1>       mov    [FFF_Valid], cl
13727                                  <1> sysfnf_11:
13728 000113C3 BE[F68A0100]           <1>       mov    esi, FindFile_Drv
13729 000113C8 EB44                   <1>       jmp    short sysfff_6
13730                                  <1> sysfff_5:
13731                                  <1>       ;mov   esi, FindFile_DirEntry
13732 000113CA B918000000             <1>       mov    ecx, 24  ; transfer length
13733 000113CF 880D[448E0100]         <1>       mov    [FFF_Valid], cl
13734                                  <1> sysfnf_12:
13735 000113D5 BF[00930100]           <1>       mov    edi, DTA ; FFF data transfer address
13736                                  <1>       ;mov   al, [esi+DirEntry_Attr] ; 11
13737 000113DA 88D8                   <1>       mov    al, bl ; File/Dir Attributes
13738 000113DC 887F17                 <1>       mov    [edi+23], bh ; Longname length (0= none)
13739 000113DF AA                     <1>       stosb
13740 000113E0 88D0                   <1>       mov    al, dl ; DL is for '?'
13741 000113E2 00F0                   <1>       add    al, dh ; DH is for '*'
13742                                  <1>       ; AL > 0 if ambiguous file name wildcards are used
13743 000113E4 AA                     <1>       stosb
13744 000113E5 8B4616                 <1>       mov    eax, [esi+DirEntry_WrtTime] ; 22
13745 000113E8 AB                     <1>         stosd      ; DirEntry_WrtTime & DirEntry_WrtDate
13746 000113E9 8B461C                 <1>         mov eax, [esi+DirEntry_FileSize] ; 28
13747 000113EC AB                     <1>         stosd
13748 000113ED 668B4614               <1>       mov    ax, [esi+DirEntry_FstClusHI] ; 20
13749 000113F1 66C1E010               <1>       shl    ax, 16
13750 000113F5 668B461A               <1>       mov    ax, [esi+DirEntry_FstClusLO] ; 26
13751 000113F9 A3[64030300]           <1>       mov    [u.r0], eax ; First Cluster
13752                                  <1>
13753                                  <1>         ;mov esi, FindFile_DirEntry
13754 000113FE E84F140000             <1>       call   get_file_name
13755                                  <1>
13756 00011403 8A0D[448E0100]         <1>       mov    cl, [FFF_Valid]
13757 00011409 BE[00930100]           <1>              mov    esi, DTA ; FFF data transfer address
13758                                  <1> sysfff_6:
13759 0001140E 8B3D[408E0100]         <1>       mov    edi, [FFF_UBuffer] ; user's buffer address (edx)
13760 00011414 E8ABFEFFFF             <1>       call   transfer_to_user_buffer
13761                                  <1>
13762 00011419 890D[64030300]         <1>       mov    [u.r0], ecx ; actual transfer count
13763 0001141F E8F1130000             <1>         call       reset_working_path
13764 00011424 E9C3C4FFFF             <1>       jmp    sysret
13765                                  <1>
13766                                  <1> sysfnf: ; <Find Next File>
13767                                  <1>       ; 16/10/2016 TRDOS 386 (TRDOS v2.0) feature only !
13768                                  <1>       ;           -derived from TRDOS v1.0, INT_21H.ASM-
13769                                  <1>       ;            ("loc_INT21h_find_next_file")
13770                                  <1>       ; TRDOS 8086 (v1.0)
13771                                  <1>         ;   07/08/2011
13772                                  <1>         ;   Find First File
13773                                  <1>       ;     INPUT:
13774                                  <1>         ;       none
13775                                  <1>       ;     MSDOS OUTPUT:
13776                                  <1>       ;         DTA: (Default address: PSP offset 80h)
13777                                  <1>       ;         Offset  Descrription
13778                                  <1>       ;         0     Reserved for use find next file
13779                                  <1>       ;         21    Attribute of file found
13780                                  <1>       ;         22    Time stamp of file
13781                                  <1>       ;         24    Date stamp of file
13782                                  <1>       ;         26    File size in bytes
13783                                  <1>       ;         30    Filename and extension (zero terminated)
13784                                  <1>       ;     If cf = 1:
13785                                  <1>       ;         Error Codes: (in AX)
13786                                  <1>       ;            18 - No more files
13787                                  <1>         ;
13788                                  <1>       ; TRDOS 386 (v2.0)
13789                                  <1>       ; 16/10/2016
13790                                  <1>       ;
13791                                  <1>         ; INPUT ->
13792                                  <1>             ;        none
13793                                  <1>       ; OUTPUT ->
13794                                  <1>       ;        EAX = 0 if CH input of 'Find First File' > 0
13795                                  <1>       ;        EAX = First cluster number of file
13796                                  <1>       ;             if CH input of 'Find First File' = 0
13797                                  <1>       ;        EDX = File parameters table/structure address
13798                                  <1>       ;
13799                                  <1>       ;          cf = 1 -> Error code in AL
13800                                  <1>       ;
13801                                  <1>       ; Modified Registers: EAX (at the return of system call)
13802                                  <1>
13803                                  <1>       ;
13804                                  <1>       ; Note: If byte [FFF_Valid] = 0
```

```
13805                                  <1>       ;      'sysfnf' will return with 'no more files' error.
13806                                  <1>       ;      If byte [FFF_Valid] = 24
13807                                  <1>       ;      'sysfnf' will return with 32 bytes basic parameters
13808                                  <1>       ;      at the address which is in EDX.
13809                                  <1>       ;      If byte [FFF_Valid] = 128
13810                                  <1>       ;      'sysfnf' will return with 128 bytes Find File
13811                                  <1>       ;      Structure/Table at the address which is in EDX.
13812                                  <1>
13813 00011429 803D[448E0100]00        <1>       cmp    byte [FFF_Valid], 0
13814 00011430 7714                    <1>       ja     short stsfnf_0
13815                                  <1>       ; 'no more files !' error
13816 00011432 B80C000000              <1>       mov    eax, ERR_NO_MORE_FILES ; 12
13817 00011437 A3[64030300]            <1>       mov    [u.r0], eax
13818 0001143C A3[C8030300]            <1>       mov    [u.error], eax
13819 00011441 E986C4FFFF              <1>       jmp    error
13820                                  <1> stsfnf_0:
13821                                  <1>       ;cmp   byte [FFF_Valid], 128
13822                                  <1>       ;je    short stsfnf_1
13823                                  <1>       ;cmp   byte [FFF_Valid], 24
13824                                  <1>       ;je    short stsfnf_1
13825                                  <1>       ;mov   [FFF_Valid], 24 ; Default
13826                                  <1> stsfnf_1:
13827 00011446 0FB61D[56820100]        <1>       movzx  ebx, byte [Current_Drv]
13828 0001144D 66891D[4A8E0100]        <1>       mov    [SWP_DRV], bx
13829 00011454 8A15[F68A0100]          <1>       mov    dl, [FindFile_Drv]
13830 0001145A 38DA                    <1>       cmp    dl, bl
13831 0001145C 750B                    <1>       jne    short stsfnf_2
13832 0001145E 86FB                    <1>       xchg   bh, bl
13833 00011460 BE00010900              <1>       mov    esi, Logical_DOSDisks
13834 00011465 01DE                    <1>       add    esi, ebx
13835 00011467 EB0D                    <1>       jmp    short sysfnf_3
13836                                  <1>
13837                                  <1> stsfnf_2:
13838 00011469 FE05[4B8E0100]          <1>       inc    byte [SWP_DRV_chg]
13839                                  <1>
13840 0001146F E85A6CFFFF              <1>       call   change_current_drive
13841 00011474 7245                    <1>       jc     short sysfnf_err_1 ; read error !
13842                                  <1>                                 ; (do not stop, because
13843                                  <1>                                 ; we don't have a
13844                                  <1>                                 ; 'no more files'
13845                                  <1>                                 ; -file not found- error,
13846                                  <1>                                 ; next sysfnf system call
13847                                  <1>                                 ; may solve the problem,
13848                                  <1>                                 ; after re-placing the disk)
13849                                  <1> sysfnf_3:
13850 00011476 A1[6C8B0100]            <1>       mov    eax, [FindFile_DirCluster]
13851 0001147B 21C0                    <1>       and    eax, eax
13852 0001147D 7550                    <1>       jnz    short sysfnf_6
13853                                  <1>
13854 0001147F 803D[55820100]02        <1>       cmp    byte [Current_FATType], 2
13855 00011486 772C                    <1>       ja     short sysfnf_err_0 ; invalid, we neeed to stop !?
13856 00011488 803D[55820100]01        <1>       cmp    byte [Current_FATType], 1
13857 0001148F 7223                    <1>       jb     short sysfnf_err_0 ; invalid, we neeed to stop !?
13858                                  <1>
13859 00011491 3805[7C890100]          <1>       cmp    byte [DirBuff_ValidData], al ; 0
13860 00011497 7608                    <1>       jna    short sysfnf_4
13861                                  <1>
13862 00011499 3B05[81890100]          <1>       cmp    eax, [DirBuff_Cluster] ; 0 ?
13863 0001149F 745E                    <1>       je     short sysfnf_9
13864                                  <1>
13865                                  <1>       ;cmp   byte [Current_Dir_Level], 0
13866                                  <1>         ;ja  short sysfnf_4
13867                                  <1>         ;jna short sysfnf_9
13868                                  <1>
13869                                  <1> sysfnf_4:
13870 000114A1 FE05[4B8E0100]          <1>       inc    byte [SWP_DRV_chg]
13871 000114A7 E805BAFFFF              <1>       call   load_FAT_root_directory
13872 000114AC 7351                    <1>       jnc    short sysfnf_9
13873                                  <1>       ; eax = error code (17, 'drv not ready or read error')
13874 000114AE EB0B                    <1>       jmp    short sysfnf_err_1 ; read error ! (no FNF stop)
13875                                  <1>                                 ; (if you want, try again,
13876                                  <1>                                 ;  after re-placing the disk)
13877                                  <1> sysfnf_5:
13878 000114B0 3C0C                    <1>       cmp    al, 12 ; 'no more files' error
13879 000114B2 7507                    <1>       jne    short sysfnf_err_1 ; (no FNF stop -sysfnf will try
13880                                  <1>                                 ;  to read the directory again,
13881                                  <1>                                 ;  if the user calls sysfnf
13882                                  <1>                                 ;  just after this error return-)
13883                                  <1>       ; (FNF stop -sysfnf will not try
13884                                  <1>       ;  to read the directory again-)
13885                                  <1>
13886                                  <1> sysfnf_err_0:
13887 000114B4 C605[448E0100]00        <1>       mov    byte [FFF_Valid], 0 ; FNF stop sign
13888                                  <1> sysfnf_err_1:
13889 000114BB A3[64030300]            <1>       mov    [u.r0], eax
13890 000114C0 A3[C8030300]            <1>       mov    [u.error], eax
13891 000114C5 E84B130000              <1>       call   reset_working_path
13892 000114CA E9FDC3FFFF              <1>       jmp    error
13893                                  <1>
13894                                  <1> sysfnf_6:
13895 000114CF 803D[7C890100]00        <1>       cmp    byte [DirBuff_ValidData], 0
13896 000114D6 7608                    <1>       jna    short sysfnf_7
13897                                  <1>
13898 000114D8 3B05[81890100]          <1>       cmp    eax, [DirBuff_Cluster]
13899 000114DE 741F                    <1>       je     short sysfnf_9
13900                                  <1>
13901                                  <1> sysfnf_7:
13902 000114E0 FE05[4B8E0100]          <1>       inc    byte [SWP_DRV_chg]
13903 000114E6 803D[55820100]01        <1>       cmp    byte [Current_FATType], 1
13904 000114ED 7309                    <1>       jnb    short sysfnf_8
13905                                  <1>
13906                                  <1>       ; Singlix (TRFS) File System
13907                                  <1>       ; (access via compatibility buffer)
13908 000114EF E885BAFFFF              <1>       call   load_FS_sub_directory
13909 000114F4 7309                    <1>       jnc    short sysfnf_9
```

```
13910                                   <1>
13911 000114F6 EBC3                     <1>       jmp    short sysfnf_err_1 ; read error (no FNF stop)
13912                                   <1>
13913                                   <1> sysfnf_8:
13914 000114F8 E83FBAFFFF               <1>       call   load_FAT_sub_directory
13915 000114FD 72BC                     <1>       jc     short sysfnf_err_1 ; read error (no FNF stop)
13916                                   <1>
13917                                   <1> sysfnf_9:
13918 000114FF E81C80FFFF               <1>       call   find_next_file
13919 00011504 72AA                     <1>       jc     short sysfnf_5
13920                                   <1>
13921 00011506 A0[458E0100]             <1>       mov    al, [FFF_Attrib]
13922                                   <1>       ;or    al, al
13923                                   <1>       ;jz    short sysfnf_10 ; 0 = No filter
13924 0001150B 34FF                     <1>       xor    al, 0FFh
13925 0001150D 20D8                     <1>       and    al, bl
13926 0001150F 75EE                     <1>       jnz    short sysfnf_9 ; search for next file until
13927                                   <1>                           ; an error return from
13928                                   <1>                           ; find_next_file procedure
13929                                   <1> sysfnf_10:
13930 00011511 0FB60D[448E0100]         <1>         movzx      ecx, byte [FFF_Valid]
13931 00011518 80F980                   <1>       cmp    cl, 128 ; complete FindFile structure/table
13932 0001151B 0F84A2FEFFFF             <1>       je     sysfnf_11
13933                                   <1>       ;cmp   cl, 24  ; basic parameters
13934                                   <1>       ;je    sysfnf_12
13935 00011521 E9AFFEFFFF               <1>       jmp    sysfnf_12
13936                                   <1>
13937                                   <1> writei:
13938                                   <1>       ; 26/10/2016
13939                                   <1>       ; 25/10/2016
13940                                   <1>       ; 23/10/2016
13941                                   <1>       ; 22/10/2016
13942                                   <1>       ; 19/10/2016 - TRDOS 386 (TRDOS v2.0)
13943                                   <1>       ; 19/05/2015 - 20/05/2015 (Retro UNIX 386 v1)
13944                                   <1>       ; 12/03/2013 - 31/07/2013 (Retro UNIX 8086 v1)
13945                                   <1>       ;
13946                                   <1>       ; Write data to file with first cluster number in EAX
13947                                   <1>       ;
13948                                   <1>       ; INPUTS ->
13949                                   <1>       ;    EAX - First cluster number of the file
13950                                   <1>       ;    EBX - File number (Open file index number)
13951                                   <1>       ;    u.count - byte count to be written
13952                                   <1>       ;    u.base - points to user buffer
13953                                   <1>       ;    u.fofp - points to dword with current file offset
13954                                   <1>       ;    i.size - file size
13955                                   <1>       ;    cdev - logical dos drive number of the file
13956                                   <1>       ; OUTPUTS ->
13957                                   <1>       ;    u.count - cleared
13958                                   <1>       ;    u.nread - accumulates total bytes passed back
13959                                   <1>       ;    i.size - new file size (if file byte offset overs file size)
13960                                   <1>       ;    u.fofp - points to u.off (with new offset value)
13961                                   <1>       ;
13962                                   <1>       ; (Retro UNIX Prototype : 11/11/2012 - 18/11/2012, UNIXCOPY.ASM)
13963                                   <1>       ; ((Modified registers: eax, edx, ebx, ecx, esi, edi, ebp))
13964                                   <1>
13965 00011526 31C9                     <1>       xor    ecx, ecx
13966 00011528 890D[8C030300]           <1>       mov    [u.nread], ecx  ; 0
13967 0001152E 66890D[C4030300]         <1>       mov    [u.pcount], cx ; 19/05/2015
13968 00011535 390D[88030300]           <1>       cmp    [u.count], ecx
13969 0001153B 7701                     <1>       ja     short writei_1
13970 0001153D C3                       <1>       retn
13971                                   <1> writei_1:
13972 0001153E 881D[048E0100]           <1>       mov    [writei.ofn], bl ; Open file number
13973 00011544 880D[3F8E0100]           <1>       mov    [setfmod], cl ; 0 ; reset 'update lm date&time' sign
13974                                   <1> dskw_0:
13975                                   <1>       ; 26/10/2016
13976                                   <1>       ; 22/10/2016, 23/10/2016, 25/10/2016
13977                                   <1>       ; 19/10/2016 - TRDOS 386 (TRDOS v2.0)
13978                                   <1>       ; 31/05/2015 - 25/07/2015 (Retro UNIX 386 v1)
13979                                   <1>       ; 26/04/2013 - 20/09/2013 (Retro UNIX 8086 v1)
13980                                   <1>       ;
13981                                   <1>       ; 01/08/2013 (mkdir w check)
13982 0001154A E8D7000000              <1>       call   mget_w
13983                                   <1>       ; eax = sector/block number
13984                                   <1>
13985 0001154F 8B1D[74030300]           <1>       mov     ebx, [u.fofp]
13986 00011555 8B13                     <1>       mov    edx, [ebx]
13987 00011557 81E2FF010000             <1>       and    edx, 1FFh  ; / test the lower 9 bits of the file offset
13988 0001155D 750C                     <1>       jnz    short dskw_1 ; / if its non-zero, branch
13989                                   <1>                         ; if zero, file offset = 0,
13990                                   <1>                               ; / 512, 1024,...(i.e., start of new block)
13991 0001155F 813D[88030300]0002-      <1>       cmp    dword [u.count], 512
13991 00011567 0000                     <1>
13992                                   <1>                          ; / if zero, is there enough data to fill
13993                                   <1>                          ; / an entire block? (i.e., no. of
13994 00011569 7337                     <1>       jnb    short dskw_2 ; / bytes to be written greater than 512.?
13995                                   <1>                        ; / Yes, branch. Don't have to read block
13996                                   <1> dskw_1: ; in as no past info. is to be saved
13997                                   <1>       ; (the entire block will be overwritten).
13998                                   <1>       ; 23/10/2016
13999                                   <1>
14000 0001156B BB[94070300]             <1>       mov    ebx, writei_buffer
14001                                   <1>       ; esi = logical dos drive description table address
14002                                   <1>       ; eax = sector number
14003                                   <1>       ; ebx = buffer address (in kernel's memory space)
14004                                   <1>       ; ecx = sector count
14005 00011570 B901000000               <1>       mov    ecx, 1
14006 00011575 E8A30D0000               <1>       call   disk_read
14007                                   <1>       ;call  dskrd  ; / no, must retain old info..
14008                                   <1>                          ; / Hence, read block 'r1' into an I/O buffer
14009 0001157A 7326                     <1>       jnc    short dskw_2
14010                                   <1>
14011                                   <1>       ; disk read error
14012 0001157C B811000000               <1>       mov    eax, 17 ; drive not ready or READ ERROR !
14013                                   <1> dskw_err: ; jump from disk write error
```

```
14014 00011581 A3[64030300]        <1>       mov   [u.r0], eax
14015 00011586 A3[C8030300]        <1>       mov   [u.error], eax
14016                              <1>
14017 0001158B 803D[3F8E0100]00    <1>       cmp   byte [setfmod], 0
14018 00011592 0F8634C3FFFF        <1>       jna   error
14019                              <1>
14020 00011598 E8AF030000          <1>       call  update_file_lmdt ; update last modif. date&time of the file
14021                              <1>       ;mov  byte [setfmod], 0
14022                              <1>
14023 0001159D E92AC3FFFF          <1>       jmp   error
14024                              <1>
14025                              <1> dskw_2: ; 3:
14026                              <1>       ; 23/10/2016
14027 000115A2 C605[E08D0100]01    <1>       mov   byte [writei.valid], 1 ; writei buffer contains valid data
14028 000115A9 56                  <1>       push  esi ; logical dos drive description table address
14029                              <1>       ; EAX (r1) = block/sector number
14030                              <1>       ;call  wslot
14031                              <1>             ; jsr r0,wslot / set write and inhibit bits in I/O queue,
14032                              <1>                     ; / proc. status=0, r5 points to 1st word of data
14033 000115AA 803D[C6030300]00    <1>       cmp   byte [u.kcall], 0
14034 000115B1 770F                <1>       ja    short dskw_4 ; zf=0 -> the caller is 'mkdir'
14035                              <1>       ;
14036 000115B3 66833D[C4030300]00  <1>       cmp   word [u.pcount], 0
14037 000115BB 7705                <1>       ja    short dskw_4
14038                              <1> dskw_3:
14039                              <1>       ; [u.base] = virtual address to transfer (as source address)
14040 000115BD E821FAFFFF          <1>       call  trans_addr_r ; translate virtual address to physical (r)
14041                              <1> dskw_4:
14042 000115C2 BB[94070300]        <1>       mov   ebx, writei_buffer
14043                              <1>       ; EBX (r5) = system (I/O) buffer address
14044 000115C7 E883FAFFFF          <1>       call  sioreg
14045                              <1>       ; ESI = file (user data) offset
14046                              <1>       ; EDI = sector (I/O) buffer offset
14047                              <1>       ; ECX = byte count
14048                              <1>       ;
14049 000115CC F3A4                <1>       rep   movsb
14050                              <1>       ; 25/07/2015
14051                              <1>       ; eax = remain bytes in buffer
14052                              <1>         ;       (check if remain bytes in the buffer > [u.pcount])
14053 000115CE 09C0                <1>       or    eax, eax
14054 000115D0 75EB                <1>       jnz   short dskw_3 ; (page end before system buffer end!)
14055                              <1>
14056                              <1>       ; 23/10/2016
14057 000115D2 B101                <1>       mov   cl, 1
14058 000115D4 5E                  <1>       pop   esi
14059 000115D5 A1[E48D0100]        <1>       mov   eax, [writei.sector]
14060                              <1>       ; esi = logical dos drive description table address
14061                              <1>       ; eax = sector number
14062                              <1>       ; ebx = writei buffer address
14063                              <1>       ; ecx = sector count
14064 000115DA E82F0D0000          <1>       call  disk_write ; / yes, write the block
14065 000115DF 7307                <1>       jnc   short dskw_5
14066                              <1>
14067 000115E1 B812000000          <1>       mov   eax, 18 ; drive not ready or WRITE ERROR !
14068 000115E6 EB99                <1>       jmp   short dskw_err
14069                              <1>
14070                              <1> dskw_5:
14071                              <1>       ; 26/10/2016
14072 000115E8 0FB61D[048E0100]    <1>       movzx ebx, byte [writei.ofn] ; open file number
14073 000115EF C0E302              <1>       shl   bl, 2 ; *4
14074 000115F2 8B83[D4910100]      <1>       mov   eax, [ebx+OF_POINTER]
14075 000115F8 3B83[FC910100]      <1>       cmp   eax, [ebx+OF_SIZE]
14076 000115FE 7606                <1>       jna   short dskw_6
14077 00011600 8983[FC910100]      <1>       mov   [ebx+OF_SIZE], eax
14078                              <1> dskw_6:
14079                              <1>       ;shr  bl, 2
14080 00011606 833D[88030300]00    <1>         cmp     dword [u.count], 0 ; / any more data to write?
14081 0001160D 760A                <1>       jna   short dskw_7
14082 0001160F A1[F48D0100]        <1>       mov   eax, [writei.fclust]
14083 00011614 E931FFFFFF          <1>       jmp   dskw_0 ; / yes, branch
14084                              <1> dskw_7:
14085                              <1>       ; update last modif. date&time of the file
14086                              <1>       ; (also updates file size as OF_SIZE)
14087 00011619 E82E030000          <1>       call  update_file_lmdt
14088                              <1>       ;mov  byte [setfmod], 0
14089                              <1>
14090                              <1>       ; 03/08/2013
14091 0001161E C605[C6030300]00    <1>       mov   byte [u.kcall], 0
14092                              <1>       ; 23/10/2016
14093                              <1>       ;mov  eax, [writei.fclust]
14094 00011625 C3                  <1>       retn
14095                              <1>
14096                              <1> mget_w:
14097                              <1>       ; 02/11/2016
14098                              <1>       ; 01/11/2016
14099                              <1>       ; 23/10/2016, 31/10/2016
14100                              <1>       ; 22/10/2016 - TRDOS 386 (TRDOS v2.0)
14101                              <1>       ; 03/06/2015 (Retro UNIX 386 v1, 'mget', u.5s)
14102                              <1>       ; 22/03/2013 - 31/07/2013 (Retro UNIX 8086 v1)
14103                              <1>       ;
14104                              <1>       ; Get existing or (allocate) a new disk block for file
14105                              <1>       ;
14106                              <1>       ; INPUTS ->
14107                              <1>       ;    [u.fofp] = file offset pointer
14108                              <1>       ;    [i.size] = file size
14109                              <1>       ;    [u.count] = byte count
14110                              <1>       ;    EAX = First cluster
14111                              <1>       ;    [cdev] = Logical dos drive number
14112                              <1>       ;    [writei.ofn] = File Number
14113                              <1>       ;                (Open file index, 0 based)
14114                              <1>       ;    ([u.off] = file offset)
14115                              <1>       ; OUTPUTS ->
14116                              <1>       ;    EAX = logical sector number
14117                              <1>       ;    ESI = Logical Dos Drive Description Table address
14118                              <1>       ;
```

```
14119                           <1>      ; Modified registers: EDX, EBX, ECX, ESI, EDI, EBP
14120                           <1>
14121 00011626 8B35[74030300]   <1>        mov     esi, [u.fofp]
14122 0001162C 8B2E             <1>      mov    ebp, [esi] ; u.off (or EBX*4+OF_POINTER)
14123                           <1>
14124 0001162E 29C9             <1>      sub    ecx, ecx
14125 00011630 8A2D[46030300]   <1>      mov    ch, [cdev]
14126                           <1>
14127 00011636 BE00010900       <1>      mov    esi, Logical_DOSDisks
14128 0001163B 01CE             <1>      add    esi, ecx
14129                           <1>
14130                           <1>      ; 31/10/2016
14131 0001163D 89C3             <1>      mov    ebx, eax ; First Cluster or FDT address
14132                           <1>
14133 0001163F 807E0300         <1>      cmp    byte [esi+LD_FATType], 0
14134 00011643 0F86DD010000     <1>      jna    mget_w_14 ; Singlix FS
14135                           <1>
14136 00011649 0FB74611         <1>      movzx  eax, word [esi+LD_BPB+BytesPerSec]
14137 0001164D 0FB65613         <1>      movzx  edx, byte [esi+LD_BPB+SecPerClust]
14138 00011651 8815[E28D0100]   <1>      mov    [writei.spc], dl  ; sectors per cluster
14139 00011657 F7E2             <1>      mul    edx
14140                           <1>      ; edx = 0
14141                           <1>      ; eax = bytes per cluster (<= 65536)
14142                           <1>
14143                           <1>      ; 02/11/2016
14144 00011659 89C1             <1>      mov    ecx, eax
14145 0001165B 48               <1>      dec    eax
14146 0001165C 66A3[E88D0100]   <1>      mov    [writei.bpc], ax
14147                           <1>
14148 00011662 89E8             <1>      mov    eax, ebp
14149 00011664 0305[88030300]   <1>      add    eax, [u.count] ; next file position
14150 0001166A 3B05[55040300]   <1>      cmp    eax, [i.size] ; <= file size ?
14151 00011670 0F86FC000000     <1>      jna    mget_w_4 ; no
14152                           <1>
14153 00011676 F7F1             <1>      div    ecx
14154 00011678 A3[F08D0100]     <1>      mov    [writei.c_index], eax ; cluster index
14155                           <1>      ; edx = byte offset in cluster (<= 65535)
14156                           <1>      ;mov   [writei.offset], dx
14157                           <1>      ;shr   dx, 9 ; / 512
14158                           <1>      ;mov   [writei.s_index], dl ; sector index in cluster (0 to spc -1)
14159                           <1>
14160 0001167D 29D2             <1>      sub    edx, edx ; 01/11/2016
14161 0001167F 8915[E48D0100]   <1>      mov    [writei.sector], edx ; 0
14162 00011685 668915[EA8D0100] <1>      mov    [writei.offset], dx  ; byte offset in cluster
14163 0001168C 8815[E38D0100]   <1>      mov    [writei.s_index], dl ; sector index in cluster (0 to spc -1)
14164                           <1>
14165 00011692 89D8             <1>      mov    eax, ebx ; First Cluster
14166                           <1>
14167                           <1>      ; is this the 1st mget_w or a next mget_w call ? (by 'writei')
14168 00011694 3815[E08D0100]   <1>      cmp    byte [writei.valid], dl ; 0
14169 0001169A 7624             <1>      jna    short mget_w_0
14170                           <1>
14171 0001169C 8815[E08D0100]   <1>      mov    byte [writei.valid], dl ; 0 ; reset ('writei' will set it)
14172                           <1>
14173 000116A2 3B05[F48D0100]   <1>      cmp    eax, [writei.fclust]
14174 000116A8 7516             <1>      jne    short mget_w_0
14175                           <1>
14176 000116AA 8A0D[46030300]   <1>      mov    cl, [cdev]
14177 000116B0 3A0D[E18D0100]   <1>      cmp    cl, [writei.drv]
14178 000116B6 7508             <1>      jne    short mget_w_0
14179                           <1>      ; [writei.l_clust] & [writei.l_index] are valid,
14180                           <1>      ;  we don't need to get last cluster & last cluster index
14181 000116B8 8B0D[008E0100]   <1>      mov    ecx, [writei.l_index]
14182 000116BE EB64             <1>      jmp    short mget_w_2
14183                           <1> mget_w_0:
14184 000116C0 A3[F48D0100]     <1>      mov    [writei.fclust], eax ; first cluster
14185                           <1>      ; edx = 0
14186 000116C5 A3[EC8D0100]     <1>      mov    [writei.cluster], eax ; first cluster ; 01/11/2016
14187 000116CA 8915[F88D0100]   <1>      mov    [writei.fs_index], edx ; 0 ; curret cluster index
14188                           <1>
14189                           <1>      ; FAT file system (FAT12, FAT16, FAT32)
14190 000116D0 E87CBEFFFF       <1>      call   get_last_cluster
14191 000116D5 0F822B010000     <1>      jc     mget_w_err ; eax = error code
14192                           <1>
14193 000116DB A3[FC8D0100]     <1>      mov    [writei.lclust], eax ; last cluster
14194                           <1>
14195 000116E0 8B0D[208C0100]   <1>      mov    ecx, [glc_index] ; last cluster index
14196 000116E6 890D[008E0100]   <1>      mov    [writei.l_index], ecx
14197                           <1>
14198 000116EC A0[048E0100]     <1>      mov    al, [writei.ofn]
14199 000116F1 FEC0             <1>      inc    al
14200 000116F3 A2[3F8E0100]     <1>      mov    [setfmod], al ; update lm date&time sign
14201                           <1>
14202                           <1> mget_w_1:
14203 000116F8 3B0D[F08D0100]   <1>      cmp    ecx, [writei.c_index]  ; last cluster index
14204 000116FE 7324             <1>      jnb    short mget_w_2 ; 01/11/2016
14205                           <1>
14206 00011700 A1[FC8D0100]     <1>      mov    eax, [writei.lclust]
14207                           <1>      ; EAX = Last cluster
14208 00011705 E855BFFFFF       <1>      call   add_new_cluster
14209 0001170A 0F82F6000000     <1>      jc     mget_w_err ; eax = error code
14210                           <1>      ; edx = 0
14211 00011710 A3[FC8D0100]     <1>      mov    [writei.lclust], eax ; (new) last cluster
14212 00011715 8B0D[008E0100]   <1>      mov    ecx, [writei.l_index]
14213 0001171B 41               <1>      inc    ecx ; add 1 to last cluster index
14214 0001171C 890D[008E0100]   <1>      mov    [writei.l_index], ecx ; current last cluster index
14215                           <1>
14216 00011722 EBD4             <1>      jmp    short mget_w_1
14217                           <1>
14218                           <1> mget_w_2:
14219 00011724 89E9             <1>      mov    ecx, ebp
14220 00011726 030D[88030300]   <1>      add    ecx, [u.count]
14221 0001172C 890D[55040300]   <1>      mov    [i.size], ecx ; save new file size
14222                           <1>      ;sub   edx, edx ; 0
14223                           <1>
```

```
14224 00011732 A0[46030300]        <1>       mov   al, [cdev]
14225 00011737 A2[E18D0100]        <1>       mov   [writei.drv], al ; physical drive number
14226                              <1>       ; edx = 0
14227 0001173C 89E8                <1>       mov   eax, ebp ; file offset
14228 0001173E 0FB70D[E88D0100]    <1>       movzx ecx, word [writei.bpc] ; bytes per cluster - 1
14229 00011745 41                  <1>       inc   ecx ; bytes per cluster
14230 00011746 F7F1                <1>       div   ecx
14231                              <1>       ; edx = byte offset in cluster (<= 65535)
14232                              <1>       ; eax = cluster index
14233 00011748 A3[F08D0100]        <1>       mov   [writei.c_index], eax
14234 0001174D 668915[EA8D0100]    <1>       mov   [writei.offset], dx
14235 00011754 66C1EA09            <1>       shr   dx, 9 ; / 512
14236 00011758 8815[E38D0100]      <1>       mov   [writei.s_index], dl ; sector index in cluster (0 to spc -1)
14237                              <1>
14238                              <1> mget_w_3:
14239 0001175E 3B05[008E0100]      <1>       cmp   eax, [writei.l_index] ; last cluster index
14240 00011764 752A                <1>       jne   short mget_w_5
14241                              <1>
14242 00011766 A3[F88D0100]        <1>       mov   [writei.fs_index], eax ; cluster index (for next check)
14243 0001176B A1[FC8D0100]        <1>       mov   eax, [writei.lclust] ; last cluster
14244 00011770 EB60                <1>       jmp   short mget_w_10
14245                              <1>
14246                              <1> mget_w_4: ; 02/11/2016
14247                              <1>       ; eax = next file position
14248 00011772 2B05[88030300]      <1>       sub   eax, [u.count] ; current file position
14249                              <1>       ; edx = 0
14250                              <1>       ; ecx = bytes per cluster
14251 00011778 F7F1                <1>       div   ecx
14252 0001177A A3[F08D0100]        <1>       mov   [writei.c_index], eax ; cluster index
14253 0001177F 668915[EA8D0100]    <1>       mov   [writei.offset], dx
14254 00011786 66C1EA09            <1>       shr   dx, 9 ; / 512
14255 0001178A 8815[E38D0100]      <1>       mov   [writei.s_index], dl ; sector index in cluster (0 to spc -1)
14256                              <1>
14257                              <1> mget_w_5:
14258 00011790 21C0                <1>       and   eax, eax ; 0 = First Cluster's index number
14259 00011792 750C                <1>       jnz   short mget_w_6
14260                              <1>
14261 00011794 A3[F88D0100]        <1>       mov   [writei.fs_index], eax ; cluster index (for next check)
14262 00011799 A1[F48D0100]        <1>       mov   eax, [writei.fclust] ; first cluster
14263 0001179E EB32                <1>       jmp   short mget_w_10
14264                              <1>
14265                              <1> mget_w_6:
14266 000117A0 3B05[F88D0100]      <1>       cmp   eax, [writei.fs_index] ; current cluster index (>0)
14267 000117A6 7507                <1>       jne   short mget_w_7
14268 000117A8 A1[EC8D0100]        <1>       mov   eax, [writei.cluster] ; current cluster
14269 000117AD EB3A                <1>       jmp   short mget_w_11
14270                              <1>
14271                              <1> mget_w_7:
14272 000117AF 89C1                <1>       mov   ecx, eax
14273 000117B1 2B0D[F88D0100]      <1>       sub   ecx, [writei.fs_index]
14274 000117B7 730D                <1>       jnc   short mget_w_8
14275                              <1>       ; get cluster by index from the first cluster
14276 000117B9 A1[F48D0100]        <1>       mov   eax, [writei.fclust]
14277 000117BE 8B0D[F08D0100]      <1>       mov   ecx, [writei.c_index]
14278 000117C4 EB05                <1>       jmp   short mget_w_9
14279                              <1>
14280                              <1> mget_w_8:
14281 000117C6 A1[EC8D0100]        <1>       mov   eax, [writei.cluster] ; beginning cluster
14282                              <1>       ; ecx = cluster sequence number after the beginning cluster
14283                              <1>       ; sub  edx, edx ; 0
14284                              <1>
14285                              <1> mget_w_9:
14286                              <1>       ; EAX = Beginning cluster
14287                              <1>       ; EDX = Sector index in disk/file section
14288                              <1>       ;      (Only for SINGLIX file system!)
14289                              <1>       ; ECX = Cluster sequence number after the beginning cluster
14290                              <1>       ; ESI = Logical DOS Drive Description Table address
14291 000117CB E895BFFFFF          <1>       call  get_cluster_by_index
14292 000117D0 7234                <1>       jc    short mget_w_err ; error code in EAX
14293                              <1>       ; EAX = Cluster number
14294                              <1> mget_w_10:
14295 000117D2 A3[EC8D0100]        <1>       mov   [writei.cluster], eax ; FDT number for Singlix File System
14296                              <1>
14297 000117D7 807E0300            <1>       cmp   byte [esi+LD_FATType], 0
14298 000117DB 7638                <1>       jna   short mget_w_13
14299                              <1>       ; 01/11/2016
14300 000117DD 8B15[F08D0100]      <1>       mov   edx, [writei.c_index]
14301 000117E3 8915[F88D0100]      <1>       mov   [writei.fs_index], edx
14302                              <1> mget_w_11:
14303 000117E9 83E802              <1>       sub   eax, 2
14304 000117EC 0FB615[E28D0100]    <1>       movzx edx, byte [writei.spc]
14305 000117F3 F7E2                <1>       mul   edx
14306                              <1>
14307 000117F5 034668              <1>       add   eax, [esi+LD_DATABegin]
14308 000117F8 8A15[E38D0100]      <1>       mov   dl, [writei.s_index]
14309 000117FE 01D0                <1>       add   eax, edx
14310                              <1> mget_w_12:
14311 00011800 A3[E48D0100]        <1>       mov   [writei.sector], eax
14312                              <1>       ;; buffer validation must be done in writei
14313                              <1>       ;;mov  byte [writei.valid], 1
14314 00011805 C3                  <1>       retn
14315                              <1>
14316                              <1> mget_w_err:
14317 00011806 A3[C8030300]        <1>       mov   [u.error], eax
14318 0001180B A3[64030300]        <1>       mov   [u.r0], eax
14319 00011810 E9B7C0FFFF          <1>       jmp   error
14320                              <1>
14321                              <1> mget_w_13:
14322                              <1>       ; EAX = FDT number (Current Section)
14323                              <1>       ; EDX = Sector index from the first section (0,1,2,3,4...)
14324 00011815 2B15[F88D0100]      <1>       sub   edx, [writei.fs_index]
14325                              <1>       ; EDX = Sector index from current section
14326 0001181B 8915[F88D0100]      <1>       mov   [writei.fs_index], edx
14327 00011821 40                  <1>       inc   eax ; the first data sector in FS disk section
14328 00011822 01D0                <1>       add   eax, edx
```

```
14329 00011824 EBDA                <1>       jmp    short mget_w_12
14330                              <1>
14331                              <1> mget_w_14:
14332 00011826 8A4E12              <1>       mov    cl, [esi+LD_FS_BytesPerSec+1]
14333 00011829 D0E9                <1>       shr    cl, 1 ;  ; 1 for 512 bytes, 4 for 2048 bytes
14334 0001182B 880D[E28D0100]      <1>       mov    [writei.spc], cl  ; sectors per cluster
14335                              <1>       ; NOTE: writei bytes per sector value is always 512 !
14336 00011831 66C705[E88D0100]00- <1>       mov    word [writei.bpc], 512
14336 00011839 02                  <1>
14337                              <1>
14338 0001183A 89E9                <1>       mov    ecx, ebp
14339 0001183C 030D[88030300]      <1>       add    ecx, [u.count] ; next file position
14340 00011842 3B0D[55040300]      <1>       cmp    ecx, [i.size] ; <= file size ?
14341 00011848 0F86C8000000        <1>       jna    mget_w_19 ; no
14342                              <1>
14343 0001184E 29D2                <1>       sub    edx, edx ; 0
14344 00011850 8915[E48D0100]      <1>       mov    [writei.sector], edx ; 0
14345 00011856 668915[EA8D0100]    <1>       mov    [writei.offset], dx  ; byte offset in cluster
14346 0001185D 8815[E38D0100]      <1>       mov    [writei.s_index], dl ; sector index in cluster (0 to spc -1)
14347                              <1>
14348 00011863 C1E909              <1>       shr    ecx, 9 ; 1 cluster = 512 bytes
14349 00011866 890D[F08D0100]      <1>       mov    [writei.c_index], ecx ; section/cluster index
14350                              <1>
14351 0001186C 89D8                <1>       mov    eax, ebx ; FDT number (First FDT address)
14352                              <1>
14353                              <1>       ; is this the 1st mget_w or a next mget_w call ? (by 'writei')
14354 0001186E 3815[E08D0100]      <1>       cmp    byte [writei.valid], dl ; 0
14355 00011874 7624                <1>       jna    short mget_w_15
14356                              <1>
14357 00011876 8815[E08D0100]      <1>       mov    byte [writei.valid], dl ; 0 ; reset ('writei' will set it)
14358                              <1>
14359 0001187C 3B05[F48D0100]      <1>       cmp    eax, [writei.fclust]
14360 00011882 7516                <1>       jne    short mget_w_15
14361                              <1>
14362 00011884 8A0D[46030300]      <1>       mov    cl, [cdev]
14363 0001188A 3A0D[E18D0100]      <1>       cmp    cl, [writei.drv]
14364 00011890 7508                <1>       jne    short mget_w_15
14365                              <1>       ; [writei.l_clust] & [writei.l_index] are valid,
14366                              <1>       ;  we don't need to get last cluster & last cluster index
14367 00011892 8B0D[008E0100]      <1>       mov    ecx, [writei.l_index]
14368 00011898 EB49                <1>       jmp    short mget_w_17
14369                              <1> mget_w_15:
14370 0001189A A3[F48D0100]        <1>       mov    [writei.fclust], eax ; first section (FDT number)
14371                              <1>       ; edx = 0
14372 0001189F 8915[EC8D0100]      <1>       mov    [writei.cluster], edx ; 0 ; current section
14373 000118A5 8915[F88D0100]      <1>       mov    [writei.fs_index], edx ; 0 ; curret section index
14374                              <1>
14375                              <1>       ; eax = FDT number (section 0 header address)
14376 000118AB E8DFBEFFFF          <1>       call   get_last_section
14377 000118B0 0F8250FFFFFF        <1>       jc     mget_w_err ; eax = error code
14378                              <1>
14379 000118B6 8915[F88D0100]      <1>       mov    [writei.fs_index], edx ; sector index in last section
14380                              <1>
14381 000118BC A3[FC8D0100]        <1>       mov    [writei.lclust], eax ; last section address
14382                              <1>
14383 000118C1 8B0D[208C0100]      <1>       mov    ecx, [glc_index] ; last section index
14384 000118C7 890D[008E0100]      <1>       mov    [writei.l_index], ecx
14385                              <1>
14386 000118CD A0[048E0100]        <1>       mov    al, [writei.ofn]
14387 000118D2 FEC0                <1>       inc    al
14388 000118D4 A2[3F8E0100]        <1>       mov    [setfmod], al ; update lm date&time sign
14389                              <1>
14390                              <1> mget_w_16:
14391                              <1>       ; edx = (existing) last section (sector) index
14392 000118D9 8B0D[F08D0100]      <1>       mov    ecx, [writei.c_index] ; final section (sector) index
14393 000118DF 29D1                <1>       sub    ecx, edx
14394 000118E1 7633                <1>       jna    short mget_w_19
14395                              <1>        ; ecx = sector count
14396                              <1> mget_w_17:
14397 000118E3 A1[FC8D0100]        <1>       mov    eax, [writei.lclust]
14398                              <1>       ; ESI = Logical dos drv desc. table address
14399                              <1>         ; EAX = Last section
14400                              <1>         ; (ECX = 0 for directory)
14401                              <1>         ; ECX = sector count (except FDT)
14402 000118E8 E865B4FFFF          <1>       call   add_new_fs_section
14403 000118ED 7312                <1>       jnc    short mget_w_18
14404                              <1>
14405                              <1>       ; If error number = 27h (insufficient disk space)
14406                              <1>       ; it is needed to check free consequent sectors
14407                              <1>       ; (1 data sector at least and +1 section header sector)
14408                              <1>
14409 000118EF 83F827              <1>       cmp    eax, 27h
14410 000118F2 0F850EFFFFFF        <1>       jne    mget_w_err ; eax = error code
14411                              <1>
14412                              <1>       ; ecx = count of free consequent sectors
14413                              <1>       ; ecx must be > 1 (1 data + 1 header sector)
14414 000118F8 49                  <1>       dec    ecx
14415 000118F9 0F8407FFFFFF        <1>       jz     mget_w_err
14416 000118FF EBE2                <1>       jmp    short mget_w_17
14417                              <1>
14418                              <1> mget_w_18:
14419 00011901 A3[FC8D0100]        <1>       mov    [writei.lclust], eax ; (new) last section
14420                              <1>       ; ecx = sector count (except section header)
14421 00011906 8B15[008E0100]      <1>       mov    edx, [writei.l_index]
14422 0001190C 01CA                <1>       add    edx, ecx ; add sector count to index
14423 0001190E 8915[008E0100]      <1>       mov    [writei.l_index], edx
14424 00011914 EBC3                <1>       jmp    short mget_w_16
14425                              <1>
14426                              <1> mget_w_19:
14427 00011916 89E9                <1>       mov    ecx, ebp
14428 00011918 030D[88030300]      <1>       add    ecx, [u.count]
14429 0001191E 890D[55040300]      <1>       mov    [i.size], ecx ; save new file size
14430                              <1>       ;sub   edx, edx ; 0
14431                              <1>
14432 00011924 A0[46030300]        <1>       mov    al, [cdev]
```

```
14433 00011929 A2[E18D0100]        <1>       mov    [writei.drv], al ; physical drive number
14434                              <1>       ; edx = 0
14435 0001192E 89E8                <1>       mov    eax, ebp ; file offset
14436 00011930 89C2                <1>       mov    edx, eax
14437                              <1>       ; 1 cluster = 512 bytes (for Singlix FS)
14438 00011932 C1E809              <1>       shr    eax, 9  ; / 512
14439 00011935 81E2FF010000        <1>       and    edx, 1FFh
14440                              <1>       ; edx = byte offset in cluster/sector (<= 511)
14441                              <1>       ; eax = section (sector/cluster) index
14442 0001193B A3[F08D0100]        <1>       mov    [writei.c_index], eax
14443 00011940 668915[EA8D0100]    <1>       mov    [writei.offset], dx
14444                              <1>       ;mov   byte [writei.s_index], 0 ; sector index in cluster
14445 00011947 E912FEFFFF          <1>       jmp    mget_w_3
14446                              <1>
14447                              <1> update_file_lmdt: ; & update file size
14448                              <1>       ; 26/10/2016
14449                              <1>       ; 24/10/2016
14450                              <1>       ; 23/10/2016
14451                              <1>       ; 22/10/2016 - TRDOS 386 (TRDOS v2.0)
14452                              <1>       ;
14453                              <1>       ; Update last modification date&time of file
14454                              <1>       ; (call from syswrite -> writei)
14455                              <1>       ; ((also updates file size)) // 26/10/2016
14456                              <1>       ;
14457                              <1>       ; INPUT:
14458                              <1>       ;      byte [setfmod] = open file number
14459                              <1>       ; OUTPUT:
14460                              <1>       ;      cf = 0 -> success !
14461                              <1>       ;      cf = 1 -> lmdt update has been failed!
14462                              <1>       ;
14463                              <1>       ; Modified registers: eax, ebx, ecx, edx, esi, edi
14464                              <1>       ;
14465                              <1>
14466                              <1>       ;cmp   byte [setfmod], 0
14467                              <1>       ;jna   short uflmdt_2 ; nothing to do
14468                              <1>
14469 0001194C 31C0                <1>       xor    eax, eax
14470                              <1>
14471 0001194E 0FB61D[3F8E0100]    <1>       movzx  ebx, byte [setfmod]
14472 00011955 FECB                <1>       dec    bl ; open file index number (0 based)
14473                              <1>
14474 00011957 8AA3[AC910100]      <1>       mov    ah, [ebx+OF_DRIVE]
14475 0001195D BE00010900          <1>       mov    esi, Logical_DOSDisks
14476 00011962 01C6                <1>       add    esi, eax
14477 00011964 C0E302              <1>       shl    bl, 2 ; *4
14478 00011967 8B8B[84910100]      <1>       mov    ecx, [ebx+OF_FCLUSTER] ; first cluster
14479 0001196D 8B93[4C920100]      <1>       mov    edx, [ebx+OF_DIRCLUSTER] ; dir cluster
14480                              <1>
14481 00011973 D0EB                <1>       shr    bl, 1 ; /2
14482 00011975 0FB7BB[EC920100]    <1>       movzx  edi, word [ebx+OF_DIRENTRY]
14483                              <1>
14484 0001197C 803D[7C890100]01    <1>       cmp    byte [DirBuff_ValidData], 1
14485 00011983 726E                <1>       jb     short uflmdt_4
14486                              <1>
14487 00011985 A0[7A890100]        <1>       mov    al, [DirBuff_DRV]
14488 0001198A 2C41                <1>       sub    al, 'A'
14489 0001198C 38E0                <1>       cmp    al, ah
14490 0001198E 7563                <1>       jne    short uflmdt_4 ; different drive
14491 00011990 8A4603              <1>       mov    al, [esi+LD_FATType]
14492 00011993 3A05[7B890100]      <1>       cmp    al, [DirBuff_FATType]
14493 00011999 755B                <1>       jne    short uflmdt_5 ; different FS type
14494 0001199B 3B15[81890100]      <1>       cmp    edx, [DirBuff_Cluster]
14495 000119A1 7553                <1>       jne    short uflmdt_5 ; different cluster
14496                              <1>
14497                              <1> uflmdt_1:
14498                              <1>       ; Directory buffer is ready here!
14499                              <1>       ; OF_FCLUSTER must be compared/verified
14500 000119A3 BE00000800          <1>       mov    esi, Directory_Buffer
14501 000119A8 66C1E705            <1>       shl    di, 5 ; dir entry index * 32
14502 000119AC 01FE                <1>       add    esi, edi ; offset
14503                              <1>       ;
14504 000119AE F6460B18            <1>       test   byte [esi+DirEntry_Attr], 18h ; Vol & Dir
14505 000119B2 750F                <1>       jnz    short uflmdt_2 ; not a valid file !
14506 000119B4 668B4614            <1>       mov    ax, [esi+DirEntry_FstClusHI]
14507 000119B8 C1E010              <1>       shl    eax, 16
14508 000119BB 668B461A            <1>       mov    ax, [esi+DirEntry_FstClusLO]
14509 000119BF 39C8                <1>       cmp    eax, ecx ; same first cluster ?
14510 000119C1 7407                <1>       je     short uflmdt_3 ; yes, it is OK !!!
14511                              <1>
14512                              <1> uflmdt_2:
14513                              <1>       ; save directory buffer if has modified/changed sign
14514                              <1>       ; (It is good to save dir buff even if the searched
14515                              <1>       ; directory entry is not found !?)
14516 000119C3 E8DEA0FFFF          <1>       call   save_directory_buffer
14517 000119C8 F9                  <1>       stc    ; update failed
14518 000119C9 C3                  <1>       retn
14519                              <1>
14520                              <1> uflmdt_3:
14521                              <1>       ; Update directory entry
14522                              <1>       ; 26/10/2016
14523 000119CA D0E3                <1>       shl    bl, 1 ; *2
14524 000119CC 8B83[FC910100]      <1>       mov    eax, [ebx+OF_SIZE] ; file size
14525 000119D2 89461C              <1>       mov    [esi+DirEntry_FileSize], eax
14526                              <1>       ;
14527 000119D5 E82EA0FFFF          <1>       call   convert_current_date_time
14528                              <1>       ; OUTPUT -> DX = Date in dos dir entry format
14529                              <1>         ;       AX = Time in dos dir entry format
14530 000119DA 66894616            <1>       mov    [esi+DirEntry_WrtTime], ax
14531 000119DE 66895618            <1>       mov    [esi+DirEntry_WrtDate], dx
14532 000119E2 66895612            <1>       mov    [esi+DirEntry_LastAccDate], dx
14533 000119E6 C605[7C890100]02    <1>       mov    byte [DirBuff_ValidData], 2
14534 000119ED E8B4A0FFFF          <1>       call   save_directory_buffer
14535 000119F2 C3                  <1>       retn
14536                              <1>
14537                              <1> uflmdt_4:
```

```
14538                                  <1>       ; Directory buffer sector read&write
14539                                  <1>       ; 23/10/2016
14540                                  <1>       ;
14541 000119F3 8A4603                  <1>       mov    al, [esi+LD_FATType]
14542                                  <1> uflmdt_5:
14543 000119F6 BB[9C090300]            <1>       mov    ebx, rw_buffer ; Common r/w sector buffer addr
14544                                  <1>
14545 000119FB 20C0                    <1>       and    al, al ; 0 = Singlix FS
14546 000119FD 0F8492000000            <1>       jz     uflmdt_11
14547                                  <1>
14548 00011A03 21D2                    <1>       and    edx, edx
14549 00011A05 7521                    <1>       jnz    short uflmdt_9
14550                                  <1>
14551 00011A07 3C02                    <1>       cmp    al, 2  ; 3 = FAT32
14552 00011A09 771A                    <1>       ja     short uflmdt_8
14553                                  <1>
14554 00011A0B 89F8                    <1>       mov    eax, edi ; directory entry index number
14555 00011A0D 66C1E804                <1>       shr    ax, 4 ; 16 entries per sector
14556 00011A11 034664                  <1>       add    eax, [esi+LD_ROOTBegin]
14557                                  <1>       ; eax = root directory sector
14558                                  <1> uflmdt_6:
14559 00011A14 50                      <1>       push   eax ; * ; disk sector address
14560 00011A15 51                      <1>       push   ecx ; first cluster
14561 00011A16 B901000000              <1>       mov    ecx, 1
14562                                  <1>       ; ecx = sector count
14563 00011A1B E8FD080000              <1>       call   disk_read
14564 00011A20 59                      <1>       pop    ecx
14565 00011A21 731A                    <1>       jnc    short uflmdt_10
14566 00011A23 58                      <1>       pop    eax ; *
14567                                  <1> uflmdt_7:
14568 00011A24 C3                      <1>       retn
14569                                  <1>
14570                                  <1> uflmdt_8:
14571 00011A25 8B5632                  <1>       mov    edx, [esi+LD_BPB+FAT32_RootFClust]
14572                                  <1> uflmdt_9:
14573 00011A28 83FA02                  <1>       cmp    edx, 2
14574 00011A2B 72F7                    <1>       jb     short uflmdt_7 ; invalid, nothing to do
14575                                  <1>
14576 00011A2D 83EA02                  <1>       sub    edx, 2
14577 00011A30 89D0                    <1>       mov    eax, edx
14578 00011A32 0FB65613                <1>       movzx  edx, byte [esi+LD_BPB+SecPerClust]
14579 00011A36 F7E2                    <1>       mul    edx
14580 00011A38 034668                  <1>       add    eax, [esi+LD_DATABegin]
14581                                  <1>       ; eax = sub directory (data) sector
14582 00011A3B EBD7                    <1>       jmp    short uflmdt_6
14583                                  <1>
14584                                  <1> uflmdt_10:
14585                                  <1>       ; Directory sector buffer is ready here!
14586                                  <1>       ; OF_FCLUSTER must be compared/verified
14587                                  <1>       ; edi = dir entry index number (<= 2047)
14588 00011A3D 6683E70F                <1>       and    di, 0Fh ; 16 entries per sector
14589 00011A41 66C1E705                <1>       shl    di, 5 ; dir entry index * 32
14590 00011A45 81C7[9C090300]          <1>       add    edi, rw_buffer
14591                                  <1>       ;
14592 00011A4B F6470B18                <1>       test   byte [edi+DirEntry_Attr], 18h ; Vol & Dir
14593 00011A4F 0F856EFFFFFF            <1>       jnz    uflmdt_2 ; not a valid file !
14594 00011A55 668B5714                <1>       mov    dx, [edi+DirEntry_FstClusHI]
14595 00011A59 C1E210                  <1>       shl    edx, 16
14596 00011A5C 668B571A                <1>       mov    dx, [edi+DirEntry_FstClusLO]
14597 00011A60 39CA                    <1>       cmp    edx, ecx ; same first cluster ?
14598 00011A62 0F855BFFFFFF            <1>       jne    uflmdt_2 ; no !?
14599                                  <1>
14600                                  <1>       ; Update directory entry
14601 00011A68 E89B9FFFFF              <1>       call   convert_current_date_time
14602                                  <1>       ; OUTPUT -> DX = Date in dos dir entry format
14603                                  <1>         ;       AX = Time in dos dir entry format
14604 00011A6D 66894716                <1>       mov    [edi+DirEntry_WrtTime], ax
14605 00011A71 66895718                <1>       mov    [edi+DirEntry_WrtDate], dx
14606 00011A75 66895712                <1>       mov    [edi+DirEntry_LastAccDate], dx
14607                                  <1>
14608 00011A79 58                      <1>       pop    eax ; *
14609                                  <1>
14610 00011A7A BB[9C090300]            <1>       mov    ebx, rw_buffer ; Common r/w sector buffer addr
14611 00011A7F B901000000              <1>       mov    ecx, 1
14612                                  <1>       ; esi = logical dos description table address
14613                                  <1>       ; eax = disk sector number/address (LBA)
14614                                  <1>       ; ecx = sector count
14615                                  <1>       ; ebx = buffer address
14616 00011A84 E885080000              <1>       call   disk_write
14617 00011A89 0F8234FFFFFF            <1>       jc     uflmdt_2
14618                                  <1>
14619                                  <1>       ; save directory buffer if has modified/changed sign
14620 00011A8F E812A0FFFF              <1>       call   save_directory_buffer
14621 00011A94 C3                      <1>       retn
14622                                  <1>
14623                                  <1> uflmdt_11:
14624                                  <1>       ; 24/10/2016
14625                                  <1>       ; Update last modification date & time of a file
14626                                  <1>       ; on a disk with Singlix File System.
14627                                  <1>       ;
14628                                  <1>       ; (Method: Read the FDT -File Description Table-
14629                                  <1>       ; sector of the file and update the lmdt data fields,
14630                                  <1>       ; then write FDT sector to the disk.
14631                                  <1>       ; /// It is easy but there is compatibility buffer
14632                                  <1>       ; method also for changing directory entry data and
14633                                  <1>       ; also there are some programming issues for Singlix
14634                                  <1>       ; file system (TRFS), which are not completed yet!)
14635                                  <1>       ;
14636                                  <1>       ; Not ready yet ! (24/10/2016)
14637                                  <1>       ; /// Temporary code for error return ! ///
14638 00011A95 31C0                    <1>       xor    eax, eax
14639 00011A97 F9                      <1>       stc
14640 00011A98 C3                      <1>       retn
14641                                  <1>
14642                                  <1> sysalloc:
```

```
14643                                  <1>       ; 14/10/2017
14644                                  <1>       ; 20/08/2017, 01/09/2017
14645                                  <1>       ; 20/02/2017, 04/03/2017, 15/05/2017
14646                                  <1>       ; 19/02/2017 - TRDOS 386 (TRDOS v2.0)
14647                                  <1>       ; (TRDOS 386 feature only!)
14648                                  <1>       ;
14649                                  <1>       ; Allocate Contiguous Memory Block/Pages (for user)
14650                                  <1>       ; (System call for DMA Buffer allocation etc.)
14651                                  <1>       ;
14652                                  <1>       ; INPUT ->
14653                                  <1>       ;     EBX = Virtual address (for user)
14654                                  <1>       ;          (Physical memory block/aperture
14655                                  <1>       ;          will be mapped to this virtual address)
14656                                  <1>       ;     ECX = Byte Count
14657                                  <1>       ;          (will be rounded up to page border)
14658                                  <1>       ;     If ECX = 0
14659                                  <1>       ;         System call will return with an error (cf=1)
14660                                  <1>       ;         but ECX will contain maximum size of
14661                                  <1>       ;         available memory aperture and physical
14662                                  <1>       ;         (beginning) address of that aperture
14663                                  <1>       ;         (which have maximum size) will be in EAX.
14664                                  <1>       ;     EDX = Upper limit of the requested physical memory
14665                                  <1>       ;          block/pages.
14666                                  <1>       ;          (The last byte address of the memory aperture
14667                                  <1>       ;           must not be equal to or above this limit.)
14668                                  <1>       ;     If EDX = 0
14669                                  <1>       ;        there is NOLIMIT !
14670                                  <1>       ;     If EDX = 0FFFFFFFFh (-1)
14671                                  <1>       ;        ESI = Lower Limit !
14672                                  <1>       ;            (Beginning of the block must not be 'less'
14673                                  <1>       ;            than this.) (Must be equal to or above...)
14674                                  <1>       ;        EDI = Upper Limit !
14675                                  <1>       ;            (End of the block must be !less! than this)
14676                                  <1>       ;            (The last byte addr of the memory aperture
14677                                  <1>       ;            must not be equal to or above this limit.)
14678                                  <1>       ;
14679                                  <1>       ; OUTPUT ->
14680                                  <1>       ;     If CF = 0
14681                                  <1>       ;     EAX = Physical address of the allocated memory block
14682                                  <1>       ;     ECX = Allocated bytes (as rounded up to page borders)
14683                                  <1>       ;     EBX = Virtual address (as rounded up)
14684                                  <1>       ;     IF CF = 1
14685                                  <1>       ;         Requested (size of) Memory block could not be
14686                                  <1>       ;         allocated to the user!
14687                                  <1>       ;     IF CF = 1 & EAX = 0 (Insufficient memory error!)
14688                                  <1>       ;        ECX = Total number of free bytes
14689                                  <1>       ;              (not size of available contiguous bytes!)
14690                                  <1>       ;     If CF = 1 & EAX > 0
14691                                  <1>       ;        there is not a memory aperture with requested size
14692                                  <1>       ;        but total free mem is not less than requested size.
14693                                  <1>       ;        EAX = Physical addr of available memory aperture
14694                                  <1>       ;             with max size
14695                                  <1>       ;              (but it doesn't fit to the conditions!)
14696                                  <1>       ;        ECX = Size of available memory aperture in bytes.
14697                                  <1>       ;     If CF = 1 -> EAX = 0FFFFFFFFh
14698                                  <1>       ;        Conditions/Parameters are wrong !
14699                                  <1>       ;        ECX is same with input value.
14700                                  <1>       ;
14701                                  <1>       ; Note:      Previously allocated pages will be deallocated if
14702                                  <1>       ;       new allocation conditions are met.
14703                                  <1>       ;
14704                                  <1>       ; Note: u.break control may be included in future versions
14705                                  <1>       ;
14706                                  <1>
14707 00011A99 31C0                   <1>       xor   eax, eax ; 0
14708                                  <1>       ; 14/10/2017
14709 00011A9B 4A                     <1>       dec   edx ; is there a limit ?
14710 00011A9C 7810                   <1>       js    short sysalloc_1 ;  0 -> 0FFFFFFFFh -> NO LIMIT
14711 00011A9E 42                     <1>       inc   edx ; > 0
14712                                  <1>       ; Check upper address limit
14713                                  <1>       ;(round up to page borders)
14714 00011A9F 81C1FF0F0000           <1>       add   ecx, PAGE_SIZE-1 ; 4095
14715 00011AA5 6681E100F0             <1>       and   cx, ~PAGE_OFF ; not 4095
14716 00011AAA 39CA                   <1>       cmp   edx, ecx ; upper limit - block size
14717 00011AAC 7224                   <1>       jb    short sysalloc_err
14718                                  <1> sysalloc_1:
14719                                  <1>       ; EAX = Beginning address (physical)
14720                                  <1>       ; EAX = 0 -> Allocate mem block from the 1st proper aperture
14721                                  <1>       ; ECX = Number of bytes to be allocated
14722 00011AAE E8C049FFFF             <1>       call  allocate_memory_block
14723 00011AB3 721D                   <1>       jc    short sysalloc_err
14724                                  <1>       ; 01/09/2017
14725 00011AB5 29C2                   <1>       sub   edx, eax ; upper limit address - beginning address
14726 00011AB7 760F                   <1>       jna   short sysalloc_3 ; begin addr not less than the limit
14727 00011AB9 39CA                   <1>       cmp   edx, ecx
14728 00011ABB 720B                   <1>       jb    short sysalloc_3 ; end address overs the limit
14729                                  <1> sysalloc_2:
14730                                  <1>       ; EAX = Beginning (physical) addr of the allocated mem block
14731                                  <1>       ; ECX = Num of allocated bytes (rounded up to page borders)
14732 00011ABD 50                     <1>       push  eax ; * ; 04/03/2017
14733                                  <1>       ; Here, requested contiquous memory pages have been allocated
14734                                  <1>       ; on Memory Allocation Table but user's page directory
14735                                  <1>       ; and page tables have not been updated yet!
14736 00011ABE 51                     <1>       push  ecx ; **
14737                                  <1>       ; ebx = virtual address (will be rounded up to page border)
14738                                  <1>       ; ecx = number of bytes to be deallocated
14739                                  <1>       ;     will be adjusted to ebx+ecx round down - ebx round up
14740 00011ABF E80A4DFFFF             <1>       call  deallocate_user_pages
14741 00011AC4 731F                   <1>       jnc   short sysalloc_4 ; EAX = Deallocated memory bytes
14742 00011AC6 59                     <1>       pop   ecx ; **
14743 00011AC7 58                     <1>       pop   eax ; *
14744                                  <1> sysalloc_3:
14745                                  <1>       ; error !
14746                                  <1>       ; restore Memory Allocation Table Content
14747 00011AC8 E8B34BFFFF             <1>       call  deallocate_memory_block
```

```
14748 00011ACD 31C0               <1>       xor   eax, eax ; 0
14749 00011ACF 48                 <1>       dec   eax ; 0FFFFFFFFh ; 15/05/2017
14750 00011AD0 EB09               <1>       jmp   short sysalloc_wrong
14751                             <1> sysalloc_err:
14752 00011AD2 8B2D[60030300]     <1>       mov   ebp, [u.usp]  ; ebp points to user's registers
14753 00011AD8 894D18             <1>       mov   [ebp+24], ecx ; return to user with ecx value
14754                             <1> sysalloc_wrong:
14755                             <1>       ; eax = 0FFFFFFFFh
14756 00011ADB A3[64030300]       <1>       mov   [u.r0], eax
14757 00011AE0 E9E7BDFFFF         <1>       jmp   error
14758                             <1> sysalloc_4:
14759 00011AE5 8B2D[60030300]     <1>       mov   ebp, [u.usp]  ; ebp points to user's registers
14760 00011AEB 894518             <1>       mov   [ebp+24], eax ; return to user with ecx value
14761 00011AEE 895D10             <1>       mov   [ebp+16], ebx ; new value of ebx (rounded up)
14762 00011AF1 89C1               <1>       mov   ecx, eax ; byte count (from 'deallocate_user_pages')
14763 00011AF3 5A                 <1>       pop   edx ; ** ; discard (another) byte count
14764 00011AF4 58                 <1>       pop   eax ; *
14765 00011AF5 A3[64030300]       <1>       mov   [u.r0], eax ; physical address
14766                             <1>
14767 00011AFA 51                 <1>       push  ecx ; 20/08/2017
14768                             <1>       ;
14769                             <1>       ; Write newly allocated contiguous (physical) pages
14770                             <1>       ; on page dir and page tables of current user/process
14771                             <1>       ; as PRESENT, USER, WRITABLE
14772                             <1>       ; (then clear allocated pages)
14773 00011AFB E8C34DFFFF         <1>       call  allocate_user_pages
14774                             <1>       ;jnc  sysret ; OK! return to process with success...
14775                             <1>
14776                             <1>       ; 20/08/2017 ('sysdma' modification)
14777 00011B00 59                 <1>       pop   ecx
14778 00011B01 A1[64030300]       <1>       mov   eax, [u.r0]   ; physical address (of the block)
14779                             <1>
14780 00011B06 721D               <1>       jc    short sysalloc_6
14781                             <1>
14782 00011B08 833D[54980100]FF   <1>       cmp   dword [dma_addr], 0FFFFFFFFh ; -1
14783 00011B0F 0F82D7BDFFFF       <1>       jb    sysret
14784                             <1>
14785 00011B15 A3[54980100]       <1>       mov   [dma_addr], eax ; save dma address for sysdma
14786 00011B1A 890D[58980100]     <1>       mov   [dma_size], ecx ; save dma buff size for sysdma
14787                             <1>
14788 00011B20 E9C7BDFFFF         <1>       jmp   sysret
14789                             <1>
14790                             <1> sysalloc_6:
14791                             <1>       ;
14792                             <1>       ; unexpected error ! insufficient memory !? conflict !?
14793                             <1>       ; (!!?there is not a free page for a new page table?!!)
14794                             <1>       ; We need to terminate process with error message !!!
14795                             <1>       ;
14796 00011B25 8B2D[60030300]     <1>       mov   ebp, [u.usp]  ; ebp points to user's registers
14797 00011B2B 8B4D18             <1>       mov   ecx, [ebp+24] ; byte count
14798                             <1>
14799                             <1>       ; 20/08/2017
14800                             <1>       ;mov  eax, [u.r0]   ; physical address (of the block)
14801                             <1>
14802                             <1>       ;
14803                             <1>       ; restore Memory Allocation Table Content
14804 00011B2E E84D4BFFFF         <1>       call  deallocate_memory_block
14805                             <1>       ;
14806 00011B33 803D[BA6F0000]03   <1>       cmp   byte [CRT_MODE], 3 ; 80x25 text mode?
14807 00011B3A 7407               <1>       je    short sysalloc_7 ; yes
14808                             <1>       ; Current mode is VGA (or CGA graphics) mode,
14809                             <1>       ; We need to return to text mode for displaying
14810                             <1>       ; error message just before 'sysexit'.
14811 00011B3C B003               <1>       mov   al, 3
14812 00011B3E E82500FFFF         <1>       call  _set_mode
14813                             <1> sysalloc_7:
14814 00011B43 BE[843B0100]       <1>       mov   esi, beep_Insufficient_Memory ; error message
14815 00011B48 E8F859FFFF         <1>       call  print_msg ; print/display the message
14816 00011B4D B801000000         <1>       mov   eax, 1 ; ax=1 is needed for 'sysexit' procedure
14817 00011B52 E91CBFFFFF         <1>       jmp   sysexit ; and terminate the process !
14818                             <1>
14819                             <1> sysdalloc:
14820                             <1>       ; 19/02/2017 - TRDOS 386 (TRDOS v2.0)
14821                             <1>       ; (TRDOS 386 feature only!)
14822                             <1>       ;
14823                             <1>       ; Deallocate Memory Block/Pages (for user)
14824                             <1>       ; (Complementary call for sysalloc.)
14825                             <1>       ;
14826                             <1>       ; INPUT ->
14827                             <1>       ;     EBX = Virtual address (for user)
14828                             <1>       ;           (will be rounded up to page border)
14829                             <1>       ;     ECX = Byte Count
14830                             <1>       ;          (will be adjusted to page borders)
14831                             <1>       ;     If ICX = 0
14832                             <1>       ;        nothing to do
14833                             <1>       ;     If EBX + ECX > User's ESP
14834                             <1>       ;        nothing to do
14835                             <1>       ;
14836                             <1>       ; Note: u.break control may be included in future versions
14837                             <1>       ;
14838                             <1>       ; OUTPUT ->
14839                             <1>       ;     If CF = 0
14840                             <1>       ;        EAX = Deallocated memory bytes
14841                             <1>       ;        EBX = Virtual address (as rounded up)
14842                             <1>       ;     IF CF = 1
14843                             <1>       ;        EAX = 0
14844                             <1>       ;
14845                             <1>       ; Note:      Main purpose of this call is to deallocate/release
14846                             <1>       ;     previously allocated (physically) contiguous memory
14847                             <1>       ;     pages but beginning (virtual) address may not be
14848                             <1>       ;     followed by physically contiguous pages. So, this
14849                             <1>       ;     system call will deallocate user's virtually
14850                             <1>       ;     contiguous memory pages. Also, there is not any
14851                             <1>       ;     objections to use this system call without sysalloc
14852                             <1>       ;     system call; only possible objection is to lost data
```

```
14853                                  <1>       ;      within user's memory space, if the beginning address
14854                                  <1>       ;      and size is not proper.
14855                                  <1>       ;
14856                                  <1>       ; Note: Empty page tables will not be deallocated!!!
14857                                  <1>       ;       (they will be deallocated at process termination)
14858                                  <1>       ;
14859                                  <1>       ; Note: When the program terminates itself or when it is
14860                                  <1>       ;      terminated by operating system kernel, all allocated
14861                                  <1>       ;      memory pages will be deallocated during termination
14862                                  <1>       ;      stage. So, 'sysdalloc' is not necessary except
14863                                  <1>       ;      forgiving memory block to other programs/processes.
14864                                  <1>       ;
14865 00011B57 8B15[5C030300]         <1>       mov    edx, [u.sp]
14866 00011B5D 8B420C                <1>       mov    eax, [edx+12] ; user's stack pointer
14867 00011B60 29C8                  <1>       sub    eax, ecx ; esp - byte count
14868 00011B62 24FC                  <1>       and    al, 0FCh ; dword alignment
14869 00011B64 39D8                  <1>       cmp    eax, ebx
14870 00011B66 7220                  <1>       jb     short sysdalloc_err ; deallocation overlaps with stack
14871                                  <1>
14872 00011B68 31C0                  <1>       xor    eax, eax
14873 00011B6A 21C9                  <1>       and    ecx, ecx
14874 00011B6C 7407                  <1>       jz     short sysdalloc_2
14875                                  <1>
14876 00011B6E E85B4CFFFF            <1>       call   deallocate_user_pages
14877 00011B73 7213                  <1>       jc     short sysdalloc_err
14878                                  <1>
14879                                  <1> sysdalloc_2:
14880 00011B75 A3[64030300]          <1>       mov    [u.r0], eax
14881 00011B7A 8B2D[60030300]        <1>       mov    ebp, [u.usp]
14882 00011B80 895D10                <1>       mov    [ebp+16], ebx ; new value of ebx
14883 00011B83 E964BDFFFF            <1>       jmp    sysret
14884                                  <1>
14885                                  <1> sysdalloc_err:
14886 00011B88 A3[64030300]          <1>       mov    [u.r0], eax ; 0
14887 00011B8D E93ABDFFFF            <1>       jmp    error
14888                                  <1>
14889                                  <1> syscalbac:
14890                                  <1>       ; SYS CALLBACK
14891                                  <1>       ; 03/08/2020
14892                                  <1>       ; 16/04/2017
14893                                  <1>       ; 14/04/2017
14894                                  <1>       ; 13/04/2017
14895                                  <1>       ; 28/02/2017
14896                                  <1>       ; 26/02/2017
14897                                  <1>       ; 24/02/2017
14898                                  <1>       ; 21/02/2017 - TRDOS 386 (TRDOS v2.0)
14899                                  <1>       ; (TRDOS 386 feature only!)
14900                                  <1>       ;
14901                                  <1>       ; Link or unlink IRQ callback service to/from user (ring 3)
14902                                  <1>       ;
14903                                  <1>       ; INPUT ->
14904                                  <1>       ;     BL = IRQ number (Hardware interrupt request number)
14905                                  <1>       ;          (0 t0 15 but IRQ 0,1,2,6,8,14,15 are prohibited)
14906                                  <1>       ;          IRQ numbers 3,4,5,7,9,10,11,12,13 are valid
14907                                  <1>       ;          (numbers >15 are invalid)
14908                                  <1>       ;
14909                                  <1>       ;     BH = 0 = Unlink IRQ (in BL) from user (ring 3) service
14910                                  <1>       ;          1 = Link IRQ by using Signal Response Byte method
14911                                  <1>       ;          2 = Link IRQ by using Callback service method
14912                                  <1>       ;          3 = Link IRQ by using Auto Increment S.R.B. method
14913                                  <1>       ;         >3 = invalid
14914                                  <1>       ;
14915                                  <1>       ;     CL = Signal Return/Response Byte value
14916                                  <1>       ;
14917                                  <1>       ;     If BH = 3, kernel will put a counter value  ; 03/08/2020
14918                                  <1>       ;               (into the S.R.B. addr)
14919                                  <1>       ;              between 0 to 255. (start value = CL+1)
14920                                  <1>       ;
14921                                  <1>       ;     NOTE: counter value, for example: even and odd numbers
14922                                  <1>       ;           may be used for -audio- DMA buffer switch
14923                                  <1>       ;           within double buffer method, etc.
14924                                  <1>       ;
14925                                  <1>       ;     EDX = Signal return (Response) byte address
14926                                  <1>       ;                           - or -
14927                                  <1>       ;           Interrupt/Callback service/routine address
14928                                  <1>       ;
14929                                  <1>       ;           (virtual address in user's memory space)
14930                                  <1>       ;
14931                                  <1>       ; OUTPUT ->
14932                                  <1>       ;     CF = 0 & EAX = 0 -> Successful setting
14933                                  <1>       ;     CF = 1 & EAX > 0 -> IRQ is prohibited or locked
14934                                  <1>       ;                  by another process
14935                                  <1>       ;            eax = ERR_PERM_DENIED -> prohibited or locked
14936                                  <1>       ;            eax = ERR_INV_PARAMETER ->
14937                                  <1>       ;                  invalid parameter/option or bad address
14938                                  <1>       ;
14939                                  <1>       ;     NOTE: Timer callbacks are set by using 'systimer'
14940                                  <1>       ;           system call (IRQ 0, PIT and IRQ 8, RTC)
14941                                  <1>       ;
14942                                  <1>       ;           Direct keyboard access is performed by using
14943                                  <1>       ;           Keyboard Interrupt (INT 32h)
14944                                  <1>       ;
14945                                  <1>       ;           It is prohibited here because:
14946                                  <1>       ;            1) Signal Response Byte method has not advantage
14947                                  <1>       ;               against INT 32h, function AH = 1. Also,
14948                                  <1>       ;               keyboard service interrupt will return with
14949                                  <1>       ;               ascii and scan codes (AL, AH) while
14950                                  <1>       ;               SRB method has only 1 byte space for ascii code
14951                                  <1>       ;               or scan code. One byte signal response is used
14952                                  <1>       ;               for ensuring very simple and very fast
14953                                  <1>       ;               virtual to physical memory address conversion
14954                                  <1>       ;               without any memory page crossover risk.
14955                                  <1>       ;               (Otherwise double page conversion or word
14956                                  <1>       ;               alignment would be needed.)
14957                                  <1>       ;            2) Badly written user code (callback code)
```

```
14958                                 <1>       ;               can prevent keyboard and timesharing functions
14959                                 <1>       ;               of the operating system via continuous and long
14960                                 <1>       ;               keyboard event handling by callback service.
14961                                 <1>       ;               (It can cause to lose immediate keystroke
14962                                 <1>       ;               response from hardware to user.)
14963                                 <1>       ;            3) If user will check any keyboard events, 'getkey'
14964                                 <1>       ;               (or 'getchar') must have more priority than other
14965                                 <1>       ;               (video etc.) events because only control ability
14966                                 <1>       ;               on a procedural infinite loop is a keyboard or
14967                                 <1>       ;               mouse event. So user can use keyboard function
14968                                 <1>       ;               at the end or at the beginning of a loop.
14969                                 <1>       ;               In this case, INT 32h is used for that purpose
14970                                 <1>       ;               and timer interrupt etc. callbacks can be used
14971                                 <1>       ;               for dynamic and synchronized data refresh/transfer
14972                                 <1>       ;               while cpu is in a static loop (without polling).
14973                                 <1>       ;               Keyboard Int callback is not more useful because
14974                                 <1>       ;               already a manual check (a key is pressed or not)
14975                                 <1>       ;               can be performed (via INT 32h, AH = 1) efficiently
14976                                 <1>       ;               in a loop to prevent a locked infinitive loop.
14977                                 <1>       ;
14978                                 <1>       ;          Disk IRQs (6,14,15) have been phohibited from ring 3
14979                                 <1>       ;          callback because, disk operations (file system services
14980                                 <1>       ;          etc.) are independent from user program, for fast disk r/w.
14981                                 <1>       ;          They are not more useful at ring 3 while they are in use
14982                                 <1>       ;          by standard diskio functions which are mandatory part of
14983                                 <1>       ;          (monolithic) OS kernel and mainprog command interpreter.
14984                                 <1>       ;          INT 33h diskio functions are enough for user level disk
14985                                 <1>       ;          r/w.
14986                                 <1>       ;
14987                                 <1>       ; TRDOS 386 - IRQ CALLBACK structures (parameters):
14988                                 <1>       ;
14989                                 <1>       ;         [u.irqlock] = 1 word, IRQ flags (0-15) that indicates
14990                                 <1>       ;                   which IRQs are locked by (that) user.
14991                                 <1>       ;                    Lock and unlock (by user) will change
14992                                 <1>       ;                   these flags or 'terminate process' (sysexit)
14993                                 <1>       ;                   will clear these flags and unlock those IRQs.
14994                                 <1>       ;
14995                                 <1>       ;                   Bit 0 is for IRQ 0 and Bit 15 is for IRQ 15
14996                                 <1>       ;
14997                                 <1>       ;         IRQ(x).owner      : 1 byte, user, [u.uno], 0 = free (unlocked)
14998                                 <1>       ;
14999                                 <1>       ;         IRQ(x).method : 1 byte for callback method & status
15000                                 <1>       ;                     0 = Signal Response Byte method
15001                                 <1>       ;                     1 = Callback service method
15002                                 <1>       ;                     >1 = invalid for current 'syscalback'.
15003                                 <1>       ;                   or(+) 80h = IRQ is in use by system (ring 0)
15004                                 <1>       ;                               function (audio etc.) or
15005                                 <1>       ;                             a device driver.
15006                                 <1>       ;                   (system function will ignore the lock/owner)
15007                                 <1>       ;
15008                                 <1>       ;         IRQ(x).srb: 1 byte, Signal Return/Response byte value
15009                                 <1>       ;                     (a fixed value by user or a counter value
15010                                 <1>       ;                    from 0 to 255, which is increased by every
15011                                 <1>       ;                    interrupt just before putting it into
15012                                 <1>       ;                    the Signal Response byte address
15013                                 <1>       ;                    (This is not used in callback serv method)
15014                                 <1>       ;
15015                                 <1>       ;         IRQ(x).addr      : 1 dword
15016                                 <1>       ;                     Signal Response Byte address (physical)
15017                                 <1>       ;                               -or-
15018                                 <1>       ;                     Callback service address (virtual)
15019                                 <1>       ;
15020                                 <1>       ;         IRQ(x).dev: 1 byte
15021                                 <1>       ;                     0 = Default device or kernel function
15022                                 <1>       ;                               -or-
15023                                 <1>       ;                     1-255 = Assigned device driver number
15024                                 <1>       ;
15025                                 <1>       ;         (x) = 3,4,5,7,9,10,11,12,13
15026                                 <1>       ;
15027                                 <1>       ;
15028                                 <1>       ;      NOTE: If user's process/program calls the kernel (INT 40h)
15029                                 <1>       ;            while it is already running in a (ring 3) callback
15030                                 <1>       ;            service, kernel will force (convert) system call to
15031                                 <1>       ;            'sysrele' (sys release). So, this feature provides
15032                                 <1>       ;            easy and simple usage of callback services without
15033                                 <1>       ;            falling into deepless <please 'callback me' then
15034                                 <1>       ;            let me 'callback you'> cycles! (User must return
15035                                 <1>       ;            from callback service by using 'sysrele' system
15036                                 <1>       ;            call, without a significant delay. Otherwise user
15037                                 <1>       ;            process/program may be late to catch the next event
15038                                 <1>       ;            within same callback purpose.
15039                                 <1>       ;
15040                                 <1>
15041 00011B92 30C0                  <1>       xor    al, al ; the caller is 'syscalbac' sign/flag
15042 00011B94 E85F180000            <1>       call   set_irq_callback_service
15043                                 <1>       ; 16/04/2017
15044 00011B99 A3[64030300]          <1>       mov    [u.r0], eax
15045 00011B9E 0F8348BDFFFF          <1>       jnc    sysret
15046 00011BA4 A3[C8030300]          <1>       mov    dword [u.error], eax
15047 00011BA9 E91EBDFFFF            <1>       jmp    error
15048                                 <1>
15049                                 <1> sysfpstat:
15050                                 <1>       ; 28/02/2017 - TRDOS 386 (TRDOS v2.0)
15051                                 <1>       ; (TRDOS 386 feature only!)
15052                                 <1>       ;
15053                                 <1>       ; Set or reset FPU registers save/restore option (for user)
15054                                 <1>       ;             (during software task switching, wswap-rswap)
15055                                 <1>       ;
15056                                 <1>       ; INPUT ->
15057                                 <1>       ;     BL = 0 -> reset
15058                                 <1>       ;     BL = 1 -> set (FPU register will be saved and restored)
15059                                 <1>       ;
15060                                 <1>       ; OUTPUT ->
15061                                 <1>       ;     cf = 0 -> no error, FPU is ready...
15062                                 <1>       ;               (EAX = 0)
```

```
15063                                  <1>       ;      Cf = 1 -> error, 80387 FPU is not ready !
15064                                  <1>       ;               (EAX = 0FFFFFFFFh)
15065                                  <1>
15066 00011BAE 31C0                   <1>       xor    eax, eax
15067 00011BB0 803D[4C8E0100]00        <1>       cmp    byte [fpready], 0
15068 00011BB7 7613                    <1>       jna    short sysfpstat_err
15069                                  <1>
15070 00011BB9 80E301                  <1>       and    bl, 1 ; use BIT 0 only !
15071 00011BBC 881D[DA030300]          <1>       mov    [u.fpsave], bl
15072 00011BC2 A3[64030300]            <1>       mov    [u.r0], eax ; 0
15073 00011BC7 E920BDFFFF              <1>       jmp    sysret
15074                                  <1>
15075                                  <1> sysfpstat_err:
15076 00011BCC 48                      <1>       dec    eax ; 0FFFFFFFFh
15077 00011BCD A3[64030300]            <1>       mov    [u.r0], eax ; -1
15078 00011BD2 E9F5BCFFFF              <1>       jmp    error
15079                                  <1>
15080                                  <1> sysdelete: ; Delete (Remove, Unlink) File
15081                                  <1>       ; 29/12/2017 (TRDOS 386 = TRDOS v2.0)
15082                                  <1>       ;
15083                                  <1>         ; INPUT ->
15084                                  <1>         ;          EBX = File name (ASCIIZ string) address
15085                                  <1>       ; OUTPUT ->
15086                                  <1>       ;          cf = 0 -> eax = 0
15087                                  <1>       ;          cf = 1 -> Error code in AL
15088                                  <1>       ;
15089                                  <1>       ; Modified Registers: EAX (at the return of system call)
15090                                  <1>       ;
15091                                  <1>
15092 00011BD7 89DE                   <1>       mov    esi, ebx
15093                                  <1>       ; file name is forced, change directory as temporary
15094                                  <1>       ;mov   ax, 1
15095                                  <1>       ;mov   [FFF.Valid], ah ; 0 ; reset
15096                                  <1>       ;call  set_working_path
15097 00011BD9 E8620B0000              <1>       call   set_working_path_x
15098 00011BDE 731D                    <1>       jnc    short sysdelete_1
15099                                  <1>
15100 00011BE0 21C0                    <1>       and    eax, eax  ; 0 -> Bad Path!
15101 00011BE2 7505                    <1>       jnz    short sysdelete_err
15102                                  <1>       ; eax = 0
15103                                  <1> sysdelete_path_err:
15104 00011BE4 B813000000              <1>       mov    eax, ERR_INV_PATH_NAME ; 'bad path name !'
15105                                  <1> sysdelete_err:
15106 00011BE9 A3[64030300]            <1>       mov    [u.r0], eax
15107 00011BEE A3[C8030300]            <1>       mov    [u.error], eax
15108 00011BF3 E81D0C0000              <1>       call   reset_working_path
15109 00011BF8 E9CFBCFFFF              <1>       jmp    error
15110                                  <1> sysdelete_1:
15111                                  <1>       ;mov   esi, FindFile_Name
15112 00011BFD 66B80018                <1>       mov    ax, 1800h ; Only files
15113 00011C01 E86B78FFFF              <1>       call   find_first_file
15114 00011C06 72E1                    <1>       jc     short sysdelete_err
15115                                  <1> sysdelete_2:
15116                                  <1>       ; check file attributes
15117                                  <1>
15118                                  <1>       ;test  bl, 17 ; system, hidden, readonly, directory
15119 00011C08 F6C307                  <1>         test bl, 7 ; system, hidden, readonly
15120 00011C0B 7407                    <1>         jz   short sysdelete_3
15121                                  <1>
15122 00011C0D B80B000000              <1>         mov  eax, ERR_FILE_ACCESS ; 11 = 'permission denied !'
15123 00011C12 EBD5                    <1>         jmp  short sysdelete_err
15124                                  <1> sysdelete_3:
15125 00011C14 6621D2                  <1>       and    dx, dx ; Ambiguous filename chars used sign (DX>0)
15126 00011C17 7407                    <1>       jz     short sysdelete_4
15127 00011C19 B81A000000              <1>       mov    eax, ERR_INV_FILE_NAME ; 26 = 'invalid file name !'
15128 00011C1E EBC9                    <1>         jmp  short sysdelete_err
15129                                  <1> sysdelete_4:
15130                                  <1>       ;mov   bh, [LongName_EntryLength]
15131 00011C20 883D[BE8B0100]          <1>       mov    [DelFile_LNEL], bh ; Long name entry length (if > 0)
15132                                  <1>       ; edi = Directory Entry Offset (DirBuff)
15133                                  <1>       ; esi = Directory Entry (FFF Structure)
15134 00011C26 E8C3A1FFFF              <1>       call   remove_file
15135 00011C2B 72BC                    <1>       jc     short sysdelete_err
15136                                  <1> sysrmdir_5:
15137 00011C2D 31C0                    <1>       xor    eax, eax ; 0
15138 00011C2F A3[64030300]            <1>       mov    [u.r0], eax
15139                                  <1>       ;mov   [u.error], eax
15140 00011C34 E8DC0B0000              <1>       call   reset_working_path
15141 00011C39 E9AEBCFFFF              <1>       jmp    sysret
15142                                  <1>
15143                                  <1>
15144                                  <1> sysrmdir: ; Remove (Unlink) Directory
15145                                  <1>       ; 19/01/2021
15146                                  <1>       ; 29/12/2017 (TRDOS 386 = TRDOS v2.0)
15147                                  <1>       ;
15148                                  <1>         ; INPUT ->
15149                                  <1>         ;          EBX = Pointer to directory name
15150                                  <1>       ; OUTPUT ->
15151                                  <1>       ;          cf = 0 -> eax = 0
15152                                  <1>       ;          cf = 1 -> Error code in AL
15153                                  <1>       ;
15154                                  <1>       ; Modified Registers: EAX (at the return of system call)
15155                                  <1>       ;
15156                                  <1>
15157                                  <1>       ; 19/01/2021
15158 00011C3E 803D[B0030300]00        <1>       cmp    byte [u.uid], 0  ; root (super user) ?
15159 00011C45 7614                    <1>       jna    short sysrmdir_0
15160                                  <1>
15161                                  <1>       ;mov   dword [u.r0], ERR_PERM_DENIED
15162 00011C47 B80B000000              <1>       mov    eax, ERR_PERM_DENIED ; ERR_NOT_SUPERUSER
15163 00011C4C A3[64030300]            <1>       mov    [u.r0], eax
15164 00011C51 A3[C8030300]            <1>       mov    [u.error], eax
15165 00011C56 E971BCFFFF              <1>       jmp    error
15166                                  <1>
15167                                  <1> sysrmdir_0:
```

```
15168 00011C5B 89DE               <1>       mov   esi, ebx
15169                             <1>       ; file name is forced, change directory as temporary
15170                             <1>       ;mov  ax, 1
15171                             <1>       ;mov  [FFF_Valid], ah ; 0 ; reset
15172                             <1>       ;call set_working_path
15173 00011C5D E8DE0A0000        <1>       call  set_working_path_x
15174 00011C62 731D               <1>       jnc   short sysrmdir_1
15175                             <1>
15176 00011C64 21C0               <1>       and   eax, eax  ; 0 -> Bad Path!
15177 00011C66 7505               <1>       jnz   short sysrmdir_err
15178                             <1>       ; eax = 0
15179                             <1> sysrmdir_not_found:
15180 00011C68 B80C000000         <1>       mov   eax, ERR_DIR_NOT_FOUND ; Directory not found !
15181                             <1> sysrmdir_err:
15182 00011C6D A3[64030300]       <1>       mov   [u.r0], eax
15183 00011C72 A3[C8030300]       <1>       mov   [u.error], eax
15184 00011C77 E8990B0000         <1>       call  reset_working_path
15185 00011C7C E94BBCFFFF         <1>       jmp   error
15186                             <1> sysrmdir_1:
15187                             <1>       ;mov  esi, FindFile_Name
15188 00011C81 66B81008           <1>       mov   ax, 0810h ; Only directories
15189 00011C85 E8E777FFFF         <1>       call  find_first_file
15190 00011C8A 7306               <1>       jnc   short sysrmdir_2
15191                             <1>
15192                             <1>       ; eax = 2 (File not found !)
15193 00011C8C 3C02               <1>       cmp   al, 2 ; ERR_NOT_FOUND
15194 00011C8E 74D8               <1>       je    short sysrmdir_not_found
15195 00011C90 EBDB               <1>       jmp   short sysrmdir_err
15196                             <1> sysrmdir_2:
15197                             <1>       ; check directory attributes
15198                             <1>
15199 00011C92 F6C307             <1>         test bl, 7 ; system, hidden, readonly
15200 00011C95 7407               <1>         jz   short sysrmdir_3
15201                             <1>
15202 00011C97 B80B000000         <1>         mov  eax, ERR_DIR_ACCESS ; 11 = 'permission denied !'
15203 00011C9C EBCF               <1>         jmp  short sysrmdir_err
15204                             <1> sysrmdir_3:
15205 00011C9E 6621D2             <1>       and   dx, dx ; Ambiguous filename chars used sign (DX>0)
15206 00011CA1 7407               <1>       jz    short sysrmdir_4
15207                             <1>       ;mov  eax, ERR_NOT_DIR ; 'not a valid directory !'
15208 00011CA3 B813000000         <1>       mov   eax, ERR_INV_PATH_NAME ; 'bad path name !'
15209 00011CA8 EBC3               <1>         jmp  short sysrmdir_err
15210                             <1> sysrmdir_4:
15211                             <1>       ;mov  bh, [LongName_EntryLength]
15212 00011CAA 883D[BE8B0100]     <1>       mov   [DelFile_LNEL], bh ; Long name entry length (if > 0)
15213                             <1>       ; edi = Directory Entry Offset (DirBuff)
15214                             <1>       ; esi = Directory Entry (FFF Structure)
15215 00011CB0 E8917EFFFF         <1>       call  delete_sub_directory
15216 00011CB5 0F8372FFFFFF       <1>       jnc   sysrmdir_5
15217                             <1> ;     jc    short sysrmdir_6
15218                             <1> ;
15219                             <1> ;     xor   eax, eax ; 0
15220                             <1> ;sysrmdir_5:
15221                             <1> ;     mov   [u.r0], eax
15222                             <1> ;     ;mov  [u.error], eax
15223                             <1> ;     call  reset_working_path
15224                             <1> ;     jmp   sysret
15225                             <1> sysrmdir_6:
15226 00011CBB A3[64030300]       <1>       mov   [u.r0], eax
15227 00011CC0 A3[C8030300]       <1>       mov   [u.error], eax
15228                             <1>
15229 00011CC5 09C0               <1>       or    eax, eax ; EAX = 0 -> Directory not empty!
15230 00011CC7 741C               <1>       jz    short sysrmdir_9
15231                             <1>
15232                             <1>       ; EAX > 0 -> Error code in AL (or AX or EAX)
15233                             <1>
15234 00011CC9 833D[72890100]01   <1>       cmp   dword [FAT_ClusterCounter], 1
15235 00011CD0 7209               <1>       jb    short sysrmdir_8
15236                             <1> sysrmdir_7:
15237                             <1>       ; ESI = Logical DOS Drive Description Table address
15238 00011CD2 66BB00FF           <1>       mov   bx, 0FF00h ; BH = FFh -> use ESI for Drive parameters
15239                             <1>                 ; BL = 0 -> Recalculate free cluster count
15240 00011CD6 E8F7B6FFFF         <1>       call  calculate_fat_freespace
15241                             <1> sysrmdir_8:
15242 00011CDB E8350B0000         <1>       call  reset_working_path
15243 00011CE0 E9E7BBFFFF         <1>       jmp   error
15244                             <1>
15245                             <1> sysrmdir_9:
15246 00011CE5 A1[72890100]       <1>       mov   eax, [FAT_ClusterCounter]
15247 00011CEA 09C0               <1>       or    eax, eax ; 0 ?
15248 00011CEC 0F847BFFFFFF       <1>       jz    sysrmdir_err
15249                             <1>       ; ESI = Logical DOS Drive Description Table address
15250 00011CF2 66BB01FF           <1>       mov   bx, 0FF01h ; BH = FFh -> use ESI for Drive parameters
15251                             <1>                 ; BL = 1 -> add free clusters
15252 00011CF6 E8D7B6FFFF         <1>       call  calculate_fat_freespace
15253 00011CFB 09C9               <1>       or    ecx, ecx
15254 00011CFD 74DC               <1>         jz   short sysrmdir_8 ; ecx = 0 -> OK
15255                             <1>       ; ecx > 0 -> Error (Recalculation is needed)
15256 00011CFF EBD1               <1>       jmp   short sysrmdir_7
15257                             <1>
15258                             <1>
15259                             <1> syschdir: ; Change Current (Working) Drive & Directory (for user)
15260                             <1>       ; 30/12/2017 (TRDOS 386 = TRDOS v2.0)
15261                             <1>       ;
15262                             <1>         ; INPUT ->
15263                             <1>         ;          EBX = Directory name (ASCIIZ string) address
15264                             <1>       ; OUTPUT ->
15265                             <1>       ;          cf = 0 -> eax = 0
15266                             <1>       ;          cf = 1 -> Error code in AL
15267                             <1>       ;
15268                             <1>       ; Modified Registers: EAX (at the return of system call)
15269                             <1>       ;
15270                             <1>       ; NOTE: If drive name is not included, only the working
15271                             <1>       ; directory (for user, not for drive/OS) will be chanded.
15272                             <1>       ; If there is a drive name (as A:, B:, C:, D: etc.)
```

```
15273                                  <1>       ; at the beginning of the ASCIIZ (directory) string,
15274                                  <1>       ; working drive and working directory (for user)
15275                                  <1>       ; will be changed together.
15276                                  <1>       ; (When the program is terminated, MainProg -internal
15277                                  <1>       ; shell- will reset working directory to the previous
15278                                  <1>       ; -current- logical drive's current directory again.)
15279                                  <1>
15280 00011D01 89DE                   <1>       mov    esi, ebx
15281                                  <1>       ; file name is not forced, change directory as temporary
15282 00011D03 31C0                   <1>       xor    eax, eax
15283                                  <1>       ;mov   [FFF_Valid], ah ; 0 ; reset
15284                                  <1>       ;call  set_working_path
15285 00011D05 E83A0A0000             <1>       call   set_working_path_xx
15286 00011D0A 731D                   <1>       jnc    short syschdir_ok
15287 00011D0C 21C0                   <1>       and    eax, eax  ; 0 -> Bad Path!
15288 00011D0E 7505                   <1>       jnz    short syschdir_err
15289                                  <1>       ; eax = 0
15290                                  <1> syschdir_not_found:
15291 00011D10 B80C000000             <1>       mov    eax, ERR_DIR_NOT_FOUND ; Directory not found !
15292                                  <1> syschdir_err:
15293 00011D15 A3[64030300]           <1>       mov    [u.r0], eax
15294 00011D1A A3[C8030300]           <1>       mov    [u.error], eax
15295 00011D1F E8F10A0000             <1>       call   reset_working_path
15296 00011D24 E9A3BBFFFF             <1>       jmp    error
15297                                  <1> syschdir_ok:
15298 00011D29 31C0                   <1>       xor    eax, eax ; 0
15299 00011D2B A3[64030300]           <1>       mov    [u.r0], eax
15300                                  <1>       ;mov   [u.error], eax
15301 00011D30 E9B7BBFFFF             <1>       jmp    sysret
15302                                  <1>
15303                                  <1>
15304                                  <1> syschmod: ; Get & Change File (or Directory) Attributes
15305                                  <1>       ; 19/01/2021
15306                                  <1>       ; 30/12/2017 (TRDOS 386 = TRDOS v2.0)
15307                                  <1>       ;
15308                                  <1>         ; INPUT ->
15309                                  <1>         ;          EBX = File/Directory (ASCIIZ) name address
15310                                  <1>       ;          CL = New attributes (if CL < 40h)
15311                                  <1>       ;          CL >= 40h -> Get File Attributes
15312                                  <1>       ; OUTPUT ->
15313                                  <1>       ;          cf = 0 -> EAX = File attributes (in AL)
15314                                  <1>       ;          cf = 1 -> Error code in AL
15315                                  <1>       ;
15316                                  <1>       ; Modified Registers: EAX (at the return of system call)
15317                                  <1>       ;
15318                                  <1>       ; MSDOS File Attributes:    (bit value of attrib byte)
15319                                  <1>       ;     ATTR_READ_ONLY      =     01h  (bit 0, 'R')
15320                                  <1>       ;     ATTR_HIDDEN  =      02h  (bit 1, 'H')
15321                                  <1>       ;     ATTR_SYSTEM  =      04h  (bit 2, 'S')
15322                                  <1>       ;     ATTR_VOLUME_ID      =      08h  (bit 3)
15323                                  <1>       ;     ATTR_DIRECTORY      =      10h  (bit 4)
15324                                  <1>       ;     ATTR_ARCHIVE =      20h  (bit 5, 'A')
15325                                  <1>       ;     ATTR_LONG_NAME      =      ATTR_READONLY |
15326                                  <1>       ;                            ATTR_HIDDEN |
15327                                  <1>       ;                            ATTR_SYSTEM |
15328                                  <1>       ;                            ATTR_VOLUME_ID
15329                                  <1>       ;     The upper two bits of attributes must be 0.
15330                                  <1>
15331                                  <1>       ; Note:      * If ATTR_DIRECTORY is set, only directory names
15332                                  <1>       ;       will be searched (and S,H,R,A attributeds of
15333                                  <1>       ;       the directory will be changed.)
15334                                  <1>       ;     * If ATTR_VOLUME_ID is set, 'syschmod' system call
15335                                  <1>       ;       will return with 'permission denied' error.
15336                                  <1>       ;     * If ATTR_DIRECTORY is not set, only file names
15337                                  <1>       ;       will be searched (and S,H,R,A attributes of the
15338                                  <1>       ;       file will be changed.)
15339                                  <1>       ;
15340                                  <1>       ; (Ony Super User can change S,H,R attributes.)
15341                                  <1>
15342 00011D35 80F940                 <1>       cmp    cl, 40h
15343 00011D38 7327                   <1>       jnb    short syschmod_0
15344                                  <1>
15345 00011D3A F6C108                 <1>       test   cl, 08h ; ATTR_VOLUME_ID
15346 00011D3D 750E                   <1>       jnz    short syschmod_perm_err
15347                                  <1>
15348                                  <1>       ; 19/01/2021
15349 00011D3F 803D[B0030300]00       <1>       cmp    byte [u.uid], 0  ; root (super user) ?
15350 00011D46 7619                   <1>       jna    short syschmod_0
15351                                  <1>
15352                                  <1>       ; Not super user..
15353 00011D48 F6C107                 <1>       test   cl, 07h      ; S,H,R attributes
15354 00011D4B 7414                   <1>       jz     short syschmod_0
15355                                  <1>
15356                                  <1> syschmod_perm_err:
15357                                  <1>       ;mov   dword [u.r0], ERR_PERM_DENIED
15358 00011D4D B80B000000             <1>       mov    eax, ERR_PERM_DENIED ; 'permission denied !'
15359 00011D52 A3[64030300]           <1>       mov    [u.r0], eax
15360 00011D57 A3[C8030300]           <1>       mov    [u.error], eax
15361 00011D5C E96BBBFFFF             <1>       jmp    error
15362                                  <1>
15363                                  <1> syschmod_0:
15364 00011D61 880D[0C8C0100]         <1>       mov    [Attributes], cl
15365 00011D67 89DE                   <1>       mov    esi, ebx
15366                                  <1>       ; file name is forced, change directory as temporary
15367                                  <1>       ;mov   ax, 1
15368                                  <1>       ;mov   [FFF_Valid], ah ; 0 ; reset
15369                                  <1>       ;call  set_working_path
15370 00011D69 E8D2090000             <1>       call   set_working_path_x
15371 00011D6E 731D                   <1>       jnc    short syschmod_1
15372 00011D70 21C0                   <1>       and    eax, eax  ; 0 -> Bad Path!
15373 00011D72 7505                   <1>       jnz    short syschmod_err
15374                                  <1>       ; eax = 0
15375                                  <1> syschmod_path_not_found:
15376 00011D74 B813000000             <1>       mov    eax, ERR_INV_PATH_NAME ; 'Bad path name !'
15377                                  <1> syschmod_err:
```

```
15378 00011D79 A3[64030300]        <1>       mov   [u.r0], eax
15379 00011D7E A3[C8030300]        <1>       mov   [u.error], eax
15380 00011D83 E88D0A0000          <1>       call  reset_working_path
15381 00011D88 E93FBBFFFF          <1>       jmp   error
15382                              <1> syschmod_1:
15383 00011D8D B008                <1>       mov   al, 08h ; Except volume labels (& long names)
15384 00011D8F A0[0C8C0100]        <1>       mov   al, [Attributes]
15385 00011D94 2410                <1>       and   al, 10h ;
15386                              <1>       ;mov  esi, FindFile_Name
15387                              <1>       ;mov  ax, 1800h ; Only files
15388                              <1>       ;mov  ax, 0810h ; Only directories
15389 00011D96 E8D676FFFF          <1>       call  find_first_file
15390                              <1>       ;jnc  short syschmod_2
15391 00011D9B 72DC                <1>       jc    short syschmod_err
15392                              <1>
15393                              <1>       ;; eax = 2 (File not found !)
15394                              <1>       ;cmp  al, 2 ; ERR_NOT_FOUND
15395                              <1>       ;jne  short syschmod_err
15396                              <1>
15397                              <1>       ;and  byte [Attributes], 10h
15398                              <1>       ;jz   short syschmod_err
15399                              <1>
15400                              <1>       ;; Directory not found !
15401                              <1>       ;mov  al, 3 ; ERR_PATH_NOT_FOUND
15402                              <1>       ;jmp  short syschmod_err
15403                              <1>
15404                              <1> syschmod_2:
15405 00011D9D 6621D2              <1>       and   dx, dx ; Ambiguous filename chars used sign (DX>0)
15406 00011DA0 7407                <1>       jz    short syschmod_3
15407 00011DA2 B81A000000          <1>       mov   eax, ERR_INV_FILE_NAME ; 'invalid file name !'
15408 00011DA7 EBD0                <1>         jmp short syschmod_err
15409                              <1> syschmod_3:
15410                              <1>       ; EDI = Directory buffer entry offset/address
15411                              <1>       ; BL = File (or Directory) Attributes
15412                              <1>       ; mov bl, [EDI+0Bh]
15413                              <1>
15414                              <1>       ; check directory attributes
15415 00011DA9 8A3D[0C8C0100]      <1>       mov   bh, [Attributes] ; new attributes
15416 00011DAF 80FF40              <1>       cmp   bh, 40h  ;>=40 -> get file/directory attributes
15417 00011DB2 732D                <1>       jnb   short syschmod_6
15418                              <1>
15419                              <1>       ; set file/directory attributes
15420 00011DB4 F6C307              <1>       test  bl, 7 ; system, hidden, readonly
15421 00011DB7 7409                <1>         jz  short syschmod_4
15422                              <1>
15423                              <1>       ; 19/01/2021
15424 00011DB9 803D[B0030300]00    <1>       cmp   byte [u.uid], 0  ; root (super user) ?
15425 00011DC0 778B                <1>       ja    short syschmod_perm_err
15426                              <1> syschmod_4:
15427 00011DC2 66817F0CA101        <1>       cmp   word [edi+DirEntry_NTRes], 01A1h ; Singlix FS
15428 00011DC8 7424                <1>       je    short syschmod_7
15429                              <1>
15430 00011DCA 887F0B              <1>       mov   [edi+0Bh], bh    ; Attributes (New!)
15431                              <1>
15432 00011DCD C605[7C890100]02    <1>       mov   byte [DirBuff_ValidData], 2 ; modified sign
15433                              <1>                                         ; to force write
15434 00011DD4 E8CD9CFFFF          <1>       call  save_directory_buffer
15435 00011DD9 729E                <1>       jc    short syschmod_err
15436                              <1>
15437                              <1> syschmod_5:
15438 00011DDB 8A1D[0C8C0100]      <1>       mov   bl, [Attributes]
15439                              <1> syschmod_6:
15440 00011DE1 0FB6C3              <1>       movzx eax, bl
15441 00011DE4 A3[64030300]        <1>       mov   [u.r0], eax
15442                              <1>       ;mov  dword [u.error], 0
15443 00011DE9 E9FEBAFFFF          <1>       jmp   sysret
15444                              <1>
15445                              <1> syschmod_7:
15446 00011DEE 29C0                <1>       sub   eax, eax
15447 00011DF0 8A25[7A890100]      <1>         mov     ah, [DirBuff_DRV]
15448 00011DF6 BE00010900          <1>       mov   esi, Logical_DOSDisks
15449 00011DFB 01C6                <1>         add     esi, eax
15450 00011DFD 807E04A1            <1>         cmp     byte [esi+LD_FSType], 0A1h
15451 00011E01 7307                <1>       jnc   short syschmod_8
15452 00011E03 B01D                <1>       mov   al, ERR_INV_DATA ; 29 = Invalid Data
15453 00011E05 E96FFFFFFF          <1>       jmp   syschmod_err
15454                              <1>
15455                              <1> syschmod_8:
15456                              <1>       ; BH = New MS-DOS File Attributes
15457 00011E0A 88F8                <1>       mov   al, bh ; File/Directory Attributes
15458 00011E0C 30E4                <1>       xor   ah, ah ; Attributes in MS-DOS format sign
15459 00011E0E E8BC87FFFF          <1>       call  change_fs_file_attributes
15460 00011E13 0F8260FFFFFF        <1>       jc    syschmod_err
15461 00011E19 EBC0                <1>       jmp   short syschmod_5
15462                              <1>
15463                              <1>
15464                              <1> sysdrive: ; Get/Set Current (Working) Drive (for user)
15465                              <1>       ; 30/12/2017 (TRDOS 386 = TRDOS v2.0)
15466                              <1>       ;
15467                              <1>         ; INPUT ->
15468                              <1>         ;          BL = Logical DOS Drive number (0=A: ... 2=C:)
15469                              <1>       ;         If BL = 0FFh -> Get Current Drive
15470                              <1>       ; OUTPUT ->
15471                              <1>       ;          cf = 0 ->
15472                              <1>       ;               AL = Current Drive number
15473                              <1>       ;               AH = The Last Logical DOS Drive no.
15474                              <1>       ;          cf = 1 -> Error code in AL
15475                              <1>       ;
15476                              <1>       ; Modified Registers: EAX (at the return of system call)
15477                              <1>       ;
15478                              <1>       ; NOTE: If the requested logical dos drive is ready,
15479                              <1>       ;     it's current current directory will be the user's
15480                              <1>       ;     (program's) current directory.
15481                              <1>       ;     (When the program is terminated, MainProg -internal
15482                              <1>       ;     shell- will reset the previous -current- logical drive
```

```
15483                                  <1>       ;       as current drive again).
15484                                  <1>
15485 00011E1B 80FBFF                  <1>       cmp    bl, 0FFh
15486 00011E1E 7435                    <1>       je     short sysdrive_ok
15487 00011E20 3A1D[22380100]          <1>       cmp    bl, [Last_DOS_DiskNo]
15488 00011E26 771E                    <1>       ja     short sysdrive_err
15489                                  <1>
15490                                  <1>       ; Save current drive and reset mode
15491                                  <1>       ; for 'reset_working_path' procedure (for MainProg)
15492 00011E28 30C0                    <1>       xor    al, al
15493 00011E2A 66A3[488E0100]          <1>       mov    [SWP_Mode], ax ; ah = 0
15494 00011E30 A0[56820100]            <1>       mov    al, [Current_Drv]
15495 00011E35 FEC4                    <1>       inc    ah ; mov ah, 1
15496 00011E37 66A3[4A8E0100]          <1>       mov    [SWP_DRV], ax
15497                                  <1>
15498 00011E3D 88DA                    <1>       mov    dl, bl
15499 00011E3F E88A62FFFF              <1>       call   change_current_drive
15500 00011E44 730F                    <1>       jnc    short sysdrive_ok
15501                                  <1> sysdrive_err:
15502 00011E46 C705[64030300]0F00-     <1>       mov    dword [u.r0], ERR_DRV_NOT_RDY ; 'drive not ready !'
15502 00011E4E 0000                    <1>
15503 00011E50 E977BAFFFF              <1>       jmp    error
15504                                  <1> sysdrive_ok:
15505 00011E55 A0[56820100]            <1>       mov    al, [Current_Drv]
15506 00011E5A 8A25[22380100]          <1>       mov    ah, [Last_DOS_DiskNo]
15507 00011E60 A3[64030300]            <1>       mov    [u.r0], eax
15508 00011E65 E982BAFFFF              <1>       jmp    sysret
15509                                  <1>
15510                                  <1>
15511                                  <1> sysdir: ; Get Current (Working) Drive & Directory (for user)
15512                                  <1>       ; 30/12/2017 (TRDOS 386 = TRDOS v2.0)
15513                                  <1>       ;
15514                                  <1>         ; INPUT ->
15515                                  <1>         ;          EBX = Current directory name buffer address
15516                                  <1>       ;            (Buffer length = 92 bytes)
15517                                  <1>       ; OUTPUT ->
15518                                  <1>       ;          AL = Current drive (0=A: .. 2=C:)
15519                                  <1>       ;         If CF = 1 -> AL = error code
15520                                  <1>       ;
15521                                  <1>       ; Modified Registers: EAX (at the return of system call)
15522                                  <1>       ;
15523                                  <1>       ; Note: Required directory name buffer length may be
15524                                  <1>       ;     <= 92 bytes for current TRDOS 386 version.
15525                                  <1>       ;     (7*12 name chars + 7 slash + 0)
15526                                  <1>
15527 00011E6A 89E5                    <1>       mov    ebp, esp
15528 00011E6C 83EC60                  <1>       sub    esp, 96
15529 00011E6F 53                      <1>       push   ebx ; User's buffer address
15530 00011E70 30D2                    <1>       xor    dl, dl ; 0 = current drive
15531 00011E72 E85391FFFF              <1>       call   get_current_directory
15532 00011E77 72CD                    <1>       jc     short sysdrive_err ; 'drive not ready !' error
15533 00011E79 89E6                    <1>       mov    esi, esp ; System's buffer address
15534 00011E7B 5F                      <1>       pop    edi  ; User's buffer address
15535                                  <1>       ; ecx = transfer (byte) count (<=92)
15536 00011E7C E843F4FFFF              <1>       call   transfer_to_user_buffer
15537 00011E81 89EC                    <1>       mov    esp, ebp
15538 00011E83 730F                    <1>       jnc    short sysdir_ok
15539                                  <1> sysdir_err:
15540 00011E85 C705[64030300]2E00-     <1>       mov    dword [u.r0], ERR_BUFFER  ; 'buffer error !'
15540 00011E8D 0000                    <1>
15541 00011E8F E938BAFFFF              <1>       jmp    error
15542                                  <1> sysdir_ok:
15543 00011E94 8A0D[56820100]          <1>       mov    cl, [Current_Drv]
15544 00011E9A 890D[64030300]          <1>       mov    [u.r0], ecx
15545 00011EA0 E947BAFFFF              <1>       jmp    sysret
15546                                  <1>
15547                                  <1>
15548                                  <1> sysldrvt: ; Get copy of Logical DOS Drive Description Table
15549                                  <1>       ; 30/12/2017 (TRDOS 386 = TRDOS v2.0)
15550                                  <1>       ;
15551                                  <1>         ; INPUT ->
15552                                  <1>       ;          BL = Logical DOS drive number (zero based)
15553                                  <1>         ;          ECX = Logical DOS drv desc table buffer addr
15554                                  <1>       ;            (Buffer length = 256 bytes)
15555                                  <1>       ; OUTPUT ->
15556                                  <1>       ;          cf = 0 ->
15557                                  <1>       ;               AL = Current Drive number
15558                                  <1>       ;               AH = The Last Logical DOS Drive no.
15559                                  <1>       ;          cf = 1 -> Error code in AL
15560                                  <1>       ;               AH = The Last Logical DOS Drive no.
15561                                  <1>       ;
15562                                  <1>       ; Modified Registers: EAX (at the return of system call)
15563                                  <1>       ;
15564                                  <1>       ; Note: Required description table buffer length is
15565                                  <1>       ;     256 bytes for current TRDOS 386 version.
15566                                  <1>
15567 00011EA5 89CF                    <1>       mov    edi, ecx ; Destination address (user space)
15568 00011EA7 88DC                    <1>       mov    ah, bl
15569 00011EA9 30C0                    <1>       xor    al, al
15570 00011EAB BE00010900              <1>       mov    esi, Logical_DOSDisks
15571 00011EB0 01C6                    <1>       add    esi, eax ; Source address (system space)
15572 00011EB2 B900010000              <1>       mov    ecx, 256 ; Byte count
15573                                  <1>                       ; Logical Dos Drv Desc Table size
15574 00011EB7 E808F4FFFF              <1>       call   transfer_to_user_buffer
15575 00011EBC 72C7                    <1>       jc     short sysdir_err
15576 00011EBE 8A2D[22380100]          <1>       mov    ch, [Last_DOS_DiskNo]
15577 00011EC4 EBCE                    <1>       jmp    short sysdir_ok
15578                                  <1>
15579                                  <1>
15580                                  <1> systime: ; Get System Date&Time
15581                                  <1>       ; 30/12/2017 (TRDOS 386 = TRDOS v2.0)
15582                                  <1>       ;
15583                                  <1>         ; INPUT -> BL =
15584                                  <1>       ;          0 = Get Date&Time in Unix/Epoch format
15585                                  <1>       ;          1 = Get Time in MSDOS format
```

```
15586                                 <1>       ;          2 = Get Date in MSDOS format
15587                                 <1>       ;          3 = Get Date&Time in MSDOS format
15588                                 <1>       ;          4 & other values =
15589                                 <1>       ;             System timer ticks will be returned
15590                                 <1>       ;             in EAX and Carry Flag will be set.
15591                                 <1>       ;             (CF will not be set if BL = 4)
15592                                 <1>       ; OUTPUT ->
15593                                 <1>       ;      For BL input = 3
15594                                 <1>       ;           EAX = Current Time (RTC)
15595                                 <1>       ;             AL = Second (DL in MSDOS)
15596                                 <1>       ;             AH = Minute (CL in MSDOS)
15597                                 <1>       ;             HW of EAX = Hour (CH in MSDOS)
15598                                 <1>       ;         EDX = Current System Date (RTC)
15599                                 <1>       ;             DL = Day (DL in MSDOS)
15600                                 <1>       ;             DH = Month (DH in MSDOS)
15601                                 <1>       ;             HW of EDX = Year (CX in MSDOS)
15602                                 <1>       ;
15603                                 <1>       ;      For BL input = 2
15604                                 <1>       ;         EAX = Current System Date (RTC)
15605                                 <1>       ;             DL = Day (DL in MSDOS)
15606                                 <1>       ;             DH = Month (DH in MSDOS)
15607                                 <1>       ;             HW of EDX = Year (CX in MSDOS)
15608                                 <1>       ;
15609                                 <1>       ;      For BL input = 1
15610                                 <1>       ;           EAX = Current Time (RTC)
15611                                 <1>       ;             AL = Second (DL in MSDOS)
15612                                 <1>       ;             AH = Minute (CL in MSDOS)
15613                                 <1>       ;             HW of EAX = Hour (CH in MSDOS)
15614                                 <1>       ;
15615                                 <1>       ;      For BL input = 0
15616                                 <1>       ;           EAX = Unix (Epoch) Time Ticks/Seconds
15617                                 <1>       ;
15618                                 <1>       ;      For BL input  = 4
15619                                 <1>       ;         EAX = System timer ticks
15620                                 <1>       ;
15621                                 <1>       ;      If CF = 1 (for other values of BL input)
15622                                 <1>       ;         EAX = System timer ticks (no error code!)
15623                                 <1>       ;
15624                                 <1>       ; Modified Registers: EAX, (EDX)
15625                                 <1>       ;             (at the return of system call)
15626                                 <1>       ;
15627                                 <1>
15628 00011EC6 20DB                  <1>       and   bl, bl
15629 00011EC8 750F                  <1>       jnz   short systime_1
15630 00011ECA E8EF57FFFF            <1>       call  epoch
15631                                 <1> systime_0:
15632 00011ECF A3[64030300]          <1>       mov   [u.r0], eax
15633 00011ED4 E913BAFFFF            <1>       jmp   sysret
15634                                 <1> systime_1:
15635 00011ED9 80FB04                <1>       cmp   bl, 4
15636 00011EDC 7211                  <1>       jb    short systime_2
15637 00011EDE A1[10820100]          <1>       mov   eax, [TIMER_LH] ; 18.2 Hz timer ticks
15638                                 <1>                             ; Note: [TIMER_LH] may be set
15639                                 <1>                             ; to wrong timer value due to
15640                                 <1>                             ; program functions.
15641                                 <1>                             ; (This value must not be
15642                                 <1>                             ; accepted as [TIMER_LH]/18.2
15643                                 <1>                             ; seconds since the midnight.)
15644 00011EE3 76EA                  <1>       jna   short systime_0
15645 00011EE5 A3[64030300]          <1>       mov   [u.r0], eax
15646 00011EEA E9DDB9FFFF            <1>       jmp   error ; cf = 1 & [u.r0] = eax = timer ticks
15647                                 <1>
15648                                 <1> systime_2:
15649                                 <1>       ;push  ebx
15650 00011EEF E82C57FFFF            <1>       call  get_rtc_date_time
15651                                 <1>       ;pop   ebx
15652 00011EF4 F6C301                <1>       test  bl, 1
15653 00011EF7 7429                  <1>       jz    short systime_4
15654 00011EF9 30E4                  <1>       xor   ah, ah
15655 00011EFB A0[5E7E0100]          <1>       mov   al, [hour]
15656 00011F00 88C2                  <1>       mov   dl, al
15657 00011F02 C1E010                <1>       shl   eax, 16
15658 00011F05 A0[627E0100]          <1>       mov   al, [second]
15659 00011F0A 8A25[607E0100]        <1>       mov   ah, [minute]
15660 00011F10 F6C302                <1>       test  bl, 2
15661 00011F13 74BA                  <1>       jz    short systime_0
15662                                 <1>       ; Check time & date match risk
15663                                 <1>       ; (23:59:59 may cause to wrong
15664                                 <1>       ; date -new day with previous date-...)
15665 00011F15 80FA17                <1>       cmp   dl, 23
15666 00011F18 7206                  <1>       jb    short systime_3
15667 00011F1A 663D3B3B              <1>       cmp   ax, (59*256)+59 ; if hour is 23:59:59
15668 00011F1E 73CF                  <1>       jnb   short systime_2 ; wait for 1 second
15669                                 <1> systime_3:
15670                                 <1>       ; eax = time
15671 00011F20 89C6                  <1>       mov   esi, eax
15672                                 <1> systime_4:
15673 00011F22 66A1[587E0100]        <1>       mov   ax, [year]
15674 00011F28 C1E010                <1>       shl   eax, 16
15675 00011F2B A0[5C7E0100]          <1>       mov   al, [day]
15676 00011F30 8A25[5A7E0100]        <1>       mov   ah, [month]
15677                                 <1>       ; eax = date
15678 00011F36 80E301                <1>       and   bl, 1
15679 00011F39 7494                  <1>       jz    short systime_0
15680 00011F3B 96                    <1>       xchg  esi, eax
15681                                 <1>       ; eax = time, esi = date
15682 00011F3C 8B2D[60030300]        <1>       mov   ebp, [u.usp]  ; EBP points to user's registers
15683                                 <1>       ; (user) edx <-- (system) esi
15684 00011F42 897514                <1>       mov   [ebp+20], esi ; return to user with EDX value
15685 00011F45 EB88                  <1>       jmp   short systime_0
15686                                 <1>
15687                                 <1>
15688                                 <1> sysstime: ; Set System Date&Time
15689                                 <1>       ; 31/12/2017
15690                                 <1>       ; 30/12/2017 (TRDOS 386 = TRDOS v2.0)
```

```
15691                                  <1>       ;
15692                                  <1>         ; INPUT -> BL =
15693                                  <1>       ;          0 = Set Date&Time in Unix/Epoch format
15694                                  <1>       ;          1 = Set Time in MSDOS format
15695                                  <1>       ;          2 = Set Date in MSDOS format
15696                                  <1>       ;          3 = Set Date&Time in MSDOS format
15697                                  <1>       ;          4 = Set System Timer (Ticks)
15698                                  <1>       ;          5 = Convert/Save current time to/as
15699                                  <1>       ;             18.2 Hz system timer ticks
15700                                  <1>       ;          6 = Convert MSDOS Date&Time to UNIX format
15701                                  <1>       ;             without setting system date&time ; (test)
15702                                  <1>       ;          7 = Convert UNIX Date&Time to MSDOS format
15703                                  <1>       ;             without setting system date&time ; (test)
15704                                  <1>       ;         8-0FFh = invalid !
15705                                  <1>       ;        ECX = Time (or Timer) value in selected format
15706                                  <1>       ;        EDX = Date value in MSDOS format if BL=2,3,6
15707                                  <1>       ;
15708                                  <1>       ; OUTPUT ->
15709                                  <1>       ;      If CF = 0 ->
15710                                  <1>       ;           EAX = Set value
15711                                  <1>       ;      If CF = 1 -> (invalid BL input)
15712                                  <1>       ;         EAX = Ticks count [TIMER_LH]
15713                                  <1>       ;
15714                                  <1>
15715 00011F47 20DB                   <1>       and    bl, bl ; 0
15716 00011F49 7511                   <1>       jnz    short sysstime_0
15717 00011F4B 89C8                   <1>       mov    eax, ecx
15718 00011F4D E8F657FFFF             <1>       call   convert_from_epoch
15719 00011F52 E8A258FFFF             <1>       call   set_rtc_date_time
15720 00011F57 E990B9FFFF             <1>       jmp    sysret
15721                                  <1> sysstime_0:
15722 00011F5C 80FB08                 <1>       cmp    bl, 8
15723 00011F5F 722D                   <1>       jb     short sysstime_1
15724                                  <1>       ; invalid input (>7)
15725 00011F61 A1[10820100]           <1>       mov    eax, [TIMER_LH] ; 18.2 Hz timer ticks
15726                                  <1>                              ; Note: [TIMER_LH] may be set
15727                                  <1>                              ; to wrong timer value due to
15728                                  <1>                              ; program functions.
15729                                  <1>                              ; (This value must not be
15730                                  <1>                              ; accepted as [TIMER_LH]/18.2
15731                                  <1>                              ; seconds since the midnight.)
15732 00011F66 A3[64030300]           <1>       mov    [u.r0], eax
15733 00011F6B E95CB9FFFF             <1>       jmp    error ; cf = 1 & [u.r0] = eax = timer ticks
15734                                  <1>
15735                                  <1> sysstime_8:
15736                                  <1>       ; BL = 7
15737 00011F70 89C8                   <1>       mov    eax, ecx ; seconds since 1/1/1970 00:00:00
15738 00011F72 E8D157FFFF             <1>       call   convert_from_epoch
15739 00011F77 30E4                   <1>       xor    ah, ah
15740 00011F79 A0[5E7E0100]           <1>       mov    al, [hour]
15741 00011F7E C1E010                 <1>       shl    eax, 16
15742 00011F81 A0[627E0100]           <1>       mov    al, [second]
15743 00011F86 8A25[607E0100]         <1>       mov    ah, [minute]
15744 00011F8C EB92                   <1>       jmp    short systime_3
15745                                  <1>
15746                                  <1> sysstime_1:
15747 00011F8E 80FB04                 <1>       cmp    bl, 4
15748 00011F91 743F                   <1>       je     short sysstime_2 ; set system timer ticks
15749 00011F93 80FB05                 <1>       cmp    bl, 5
15750 00011F96 754B                   <1>       jne    short sysstime_4
15751                                  <1>       ; convert current time to system timer ticks (18.2Hz)
15752 00011F98 E88356FFFF             <1>       call   get_rtc_date_time
15753 00011F9D 0FB60D[5E7E0100]       <1>       movzx  ecx, byte [hour]
15754 00011FA4 B8100E0000             <1>       mov    eax, 60*60 ; 1 hour = 3600 seconds
15755 00011FA9 F7E1                   <1>       mul    ecx
15756 00011FAB 89C3                   <1>       mov    ebx, eax
15757 00011FAD B13C                   <1>       mov    cl, 60  ; 1 minute = 60 seconds
15758 00011FAF 0FB605[607E0100]       <1>       movzx  eax, byte [minute]
15759 00011FB6 F7E1                   <1>       mul    ecx
15760 00011FB8 01D8                   <1>       add    eax, ebx
15761 00011FBA 8A0D[627E0100]         <1>       mov    cl, [second]
15762 00011FC0 01C8                   <1>       add    eax, ecx
15763 00011FC2 B1B6                   <1>       mov    cl, 182
15764 00011FC4 F7E1                   <1>       mul    ecx
15765 00011FC6 83C009                 <1>       add    eax, 9
15766 00011FC9 83D200                 <1>       adc    edx, 0
15767 00011FCC B10A                   <1>       mov    cl, 10
15768 00011FCE F7F1                   <1>       div    ecx
15769                                  <1>       ; eax = ((182*seconds)+9)/10
15770 00011FD0 89C1                   <1>       mov    ecx, eax
15771                                  <1> sysstime_2:
15772 00011FD2 890D[10820100]         <1>       mov    [TIMER_LH], ecx ; 18.2 * seconds
15773                                  <1> sysstime_3:
15774 00011FD8 890D[64030300]         <1>       mov    [u.r0], ecx
15775 00011FDE E909B9FFFF             <1>       jmp    sysret
15776                                  <1> sysstime_4:
15777 00011FE3 80FB06                 <1>       cmp    bl, 6
15778 00011FE6 7788                   <1>       ja     short sysstime_8
15779                                  <1>
15780 00011FE8 890D[64030300]         <1>       mov    [u.r0], ecx
15781                                  <1>
15782 00011FEE 880D[627E0100]         <1>       mov    [second], cl
15783 00011FF4 882D[607E0100]         <1>       mov    [minute], ch
15784 00011FFA C1E910                 <1>       shr    ecx, 16
15785 00011FFD 880D[5E7E0100]         <1>       mov    [hour], cl
15786                                  <1>       ; BL = 1,2,3,6
15787 00012003 80FB01                 <1>       cmp    bl, 1
15788 00012006 762A                   <1>       jna    short sysstime_5
15789                                  <1>       ; BL = 2,3,6
15790 00012008 8815[5C7E0100]         <1>       mov    [day], dl
15791 0001200E 8835[5A7E0100]         <1>       mov    [month], dh
15792 00012014 C1EA10                 <1>       shr    edx, 16
15793 00012017 668915[587E0100]       <1>       mov    [year], dx
15794 0001201E 80E303                 <1>       and    bl, 3
15795 00012021 742D                   <1>       jz     short sysstime_7 ; 6
```

```
15796                                  <1>       ; BL = 2,3
15797 00012023 F6C301                  <1>       test  bl, 1
15798 00012026 7419                    <1>       jz    short sysstime_6 ; 2
15799                                  <1>       ; BL = 3
15800 00012028 E8CC57FFFF              <1>       call  set_rtc_date_time
15801 0001202D E9BAB8FFFF              <1>       jmp   sysret
15802                                  <1> sysstime_5:
15803                                  <1>       ; BL = 1
15804 00012032 E80358FFFF              <1>       call  set_time_bcd
15805 00012037 E8934AFFFF              <1>       call  set_rtc_time
15806 0001203C E9ABB8FFFF              <1>       jmp   sysret
15807                                  <1> sysstime_6:
15808                                  <1>       ; BL = 2
15809 00012041 E8C757FFFF              <1>       call  set_date_bcd
15810 00012046 E8F34AFFFF              <1>       call  set_rtc_date
15811 0001204B E99CB8FFFF              <1>       jmp   sysret
15812                                  <1> sysstime_7:
15813                                  <1>       ; BL = 6
15814                                  <1>       ; [year], [month], [day],
15815                                  <1>       ; [hour], [minute], [second]
15816 00012050 E86E56FFFF              <1>       call  convert_to_epoch
15817 00012055 89C1                    <1>       mov   ecx, eax ; seconds since 1/1/1970 00:00:00
15818 00012057 E97CFFFFFF              <1>       jmp   sysstime_3
15819                                  <1>
15820                                  <1> sysrename: ; Rename File (or Directory)
15821                                  <1>       ; 19/01/2021
15822                                  <1>       ; 31/12/2017 (TRDOS 386 = TRDOS v2.0)
15823                                  <1>       ;
15824                                  <1>         ; INPUT ->
15825                                  <1>         ;          EBX = File/Directory (ASCIIZ) name address
15826                                  <1>       ;         ECX = New name (in same dir, no path name)
15827                                  <1>       ; OUTPUT ->
15828                                  <1>       ;          cf = 0 -> EAX = 0
15829                                  <1>       ;          cf = 1 -> Error code in AL
15830                                  <1>
15831                                  <1>       ; 19/01/2021
15832 0001205C 803D[B0030300]00        <1>       cmp   byte [u.uid], 0  ; root (super user) ?
15833 00012063 7614                    <1>       jna   short sysrename_0
15834                                  <1>
15835                                  <1> sysrename_perm_err:
15836                                  <1>       ;mov  dword [u.r0], ERR_PERM_DENIED
15837 00012065 B80B000000              <1>       mov   eax, ERR_PERM_DENIED ; 'permission denied !'
15838 0001206A A3[64030300]            <1>       mov   [u.r0], eax
15839 0001206F A3[C8030300]            <1>       mov   [u.error], eax
15840 00012074 E953B8FFFF              <1>       jmp   error
15841                                  <1>
15842                                  <1> sysrename_0:
15843 00012079 51                      <1>       push  ecx ; new file name address (in user space)
15844 0001207A 89DE                    <1>       mov   esi, ebx
15845                                  <1>       ; file name is forced, change directory as temporary
15846                                  <1>       ;mov  ax, 1
15847                                  <1>       ;mov  [FFF_Valid], ah ; 0 ; reset
15848                                  <1>       ;call set_working_path
15849 0001207C E8BF060000              <1>       call  set_working_path_x
15850 00012081 731E                    <1>       jnc   short sysrename_1
15851 00012083 21C0                    <1>       and   eax, eax  ; 0 -> Bad Path!
15852 00012085 7505                    <1>       jnz   short sysrename_err
15853                                  <1>       ; eax = 0
15854                                  <1> sysrename_path_not_found:
15855 00012087 B813000000              <1>       mov   eax, ERR_INV_PATH_NAME ; 'Bad path name !'
15856                                  <1> sysrename_err:
15857 0001208C 59                      <1>       pop   ecx ; new file name address (in user space)
15858                                  <1> sysrename_error:
15859 0001208D A3[64030300]            <1>       mov   [u.r0], eax
15860 00012092 A3[C8030300]            <1>       mov   [u.error], eax
15861 00012097 E879070000              <1>       call  reset_working_path
15862 0001209C E92BB8FFFF              <1>       jmp   error
15863                                  <1> sysrename_1:
15864 000120A1 B008                    <1>       mov   al, 08h ; Except volume labels (& long names)
15865 000120A3 A0[0C8C0100]            <1>       mov   al, [Attributes]
15866 000120A8 2410                    <1>       and   al, 10h ;
15867                                  <1>       ;mov  esi, FindFile Name
15868                                  <1>       ;mov  ax, 1800h ; Only files
15869                                  <1>       ;mov  ax, 0810h ; Only directories
15870 000120AA 66B80008                <1>       mov   ax, 0800h ; Find File or Directory
15871 000120AE E8BE73FFFF              <1>       call  find_first_file
15872                                  <1>       ;jnc  short sysrename_2
15873 000120B3 72D7                    <1>       jc    short sysrename_err
15874                                  <1> sysrename_2:
15875                                  <1>       ; ESI = Directory Entry (FindFile_DirEntry) Location
15876                                  <1>       ; EDI = Directory Buffer Directory Entry Location
15877                                  <1>       ; EAX = File Size
15878                                  <1>       ;  BL = Attributes of The File/Directory
15879                                  <1>       ;  BH = Long Name Yes/No Status (>0 is YES)
15880                                  <1>       ;  DX > 0 : Ambiguous filename chars are used
15881                                  <1>
15882 000120B5 6621D2                  <1>       and   dx, dx ; Ambiguous filename chars used sign (DX>0)
15883 000120B8 7407                    <1>       jz    short sysrename_3
15884 000120BA B81A000000              <1>       mov   eax, ERR_INV_FILE_NAME ; 'invalid file name !'
15885 000120BF EBCB                    <1>         jmp short sysrename_err
15886                                  <1> sysrename_3:
15887                                  <1>       ; EDI = Directory buffer entry offset/address
15888                                  <1>       ; BL = File (or Directory) Attributes
15889                                  <1>       ; mov bl, [EDI+0Bh]
15890                                  <1>
15891 000120C1 5A                      <1>       pop   edx ; new file name address (in user space)
15892                                  <1>
15893                                  <1>       ; check file/directory attributes
15894 000120C2 F6C307                  <1>       test  bl, 7 ; system, hidden, readonly
15895 000120C5 759E                    <1>         jnz short sysrename_perm_err
15896                                  <1> sysrename_4:
15897 000120C7 66817F0CA101            <1>       cmp   word [edi+DirEntry_NTRes], 01A1h ; Singlix FS
15898 000120CD 7496                    <1>       je    short sysrename_perm_err ; -temporary!-
15899                                  <1>
15900                                  <1>       ; save old file name & file info (FFF structure)
```

```
15901 000120CF BE[F68A0100]        <1>       mov   esi, FindFile_Drv
15902 000120D4 BF[3C8C0100]        <1>       mov   edi, SourceFile_Drv
15903 000120D9 B920000000          <1>       mov   ecx, 128/4
15904 000120DE F3A5                <1>       rep   movsd
15905                              <1>
15906 000120E0 89D6                <1>       mov   esi, edx ; new file name address (in user space)
15907 000120E2 BF[BC8C0100]        <1>       mov   edi, DestinationFile_Drv
15908 000120E7 E85695FFFF          <1>       call  parse_path_name
15909 000120EC 729F                <1>       jc    short sysrename_error ; eax = 1 (Bad file name)
15910                              <1>
15911                              <1>       ; same drive ?
15912 000120EE A0[F68A0100]        <1>       mov   al, [FindFile_Drv]
15913 000120F3 3A05[BC8C0100]      <1>       cmp   al, [DestinationFile_Drv]
15914                              <1>       ;jne  short sysrename_perm_err ; Permission denied
15915 000120F9 7509                <1>       jne   short sysrename_5 ; Bad file name
15916                              <1>
15917                              <1>       ; no path name !? (rename file in same directory)
15918 000120FB 803D[BD8C0100]20    <1>       cmp   byte [DestinationFile_Directory], 20h
15919 00012102 7607                <1>       jna   short sysrename_6
15920                              <1> sysrename_5:
15921 00012104 B801000000          <1>       mov   eax, ERR_BAD_CMD_ARG ; 1 = Bad file name
15922                              <1>                                  ; (Bad argument)
15923 00012109 EB82                <1>       jmp   short sysrename_error
15924                              <1> sysrename_6:
15925 0001210B 803D[FE8C0100]20    <1>       cmp   byte [DestinationFile_Name], 20h
15926 00012112 76F0                <1>       jna   short sysrename_5
15927                              <1>
15928 00012114 BE[FE8C0100]        <1>       mov   esi, DestinationFile_Name
15929 00012119 E81177FFFF          <1>       call  check_filename ; is it a valid msdos file name?
15930 0001211E 0F8269FFFFFF        <1>       jc    sysrename_error ; 26 = ERR_INV_FILE_NAME
15931                              <1>
15932                              <1>       ;mov  esi, DestinationFile_Name
15933 00012124 66B80008            <1>       mov   ax, 0800h ; Find File or Directory
15934 00012128 E84473FFFF          <1>       call  find_first_file
15935 0001212D 720A                <1>       jc    short sysrename_7
15936                              <1>
15937 0001212F B80E000000          <1>       mov   eax, ERR_FILE_EXISTS  ; file already exists !
15938 00012134 E954FFFFFF          <1>       jmp   sysrename_error
15939                              <1> sysrename_7:
15940                              <1>       ; eax = 2 (File not found !)
15941 00012139 3C02                <1>       cmp   al, 2 ; ERR_NOT_FOUND
15942 0001213B 0F854CFFFFFF        <1>       jne   sysrename_error
15943                              <1>
15944                              <1>       ; 31/12/2017
15945                              <1>       ; Following code is also part of 'rename_file' in
15946                              <1>       ; 'trdosk3.s' (MainProg's 'rename' command) ; 13/11/2017
15947 00012141 BE[FE8C0100]        <1>       mov   esi, DestinationFile_Name ; (Rename_NewName)
15948 00012146 668B0D[B68C0100]    <1>       mov   cx, [SourceFile_DirEntryNumber]
15949 0001214D 66A1[A28C0100]      <1>       mov   ax, [SourceFile_DirEntry+20] ; First Cluster, HW
15950 00012153 C1E010              <1>       shl   eax, 16
15951 00012156 66A1[A88C0100]      <1>       mov   ax, [SourceFile_DirEntry+26] ; First Cluster, LW
15952 0001215C 0FB61D[8B8C0100]    <1>       movzx ebx, byte [SourceFile_LongNameEntryLength]
15953 00012163 E8229DFFFF          <1>       call  rename_directory_entry
15954 00012168 0F821FFFFFFF        <1>       jc    sysrename_error
15955                              <1>       ;xor  eax, eax
15956 0001216E A3[64030300]        <1>       mov   [u.r0], eax ; 0
15957                              <1>       ;mov  [u.error], eax
15958 00012173 E89D060000          <1>       call  reset_working_path
15959 00012178 E96FB7FFFF          <1>       jmp   sysret
15960                              <1>
15961                              <1> sysmem: ; Get Total&Free Memory amount
15962                              <1>       ; 31/12/2017 (TRDOS 386 = TRDOS v2.0)
15963                              <1>       ;
15964                              <1>         ; INPUT ->
15965                              <1>       ;     none
15966                              <1>       ; OUTPUT ->
15967                              <1>       ;     EAX = Total memory count (in bytes)
15968                              <1>       ;     EBX = Virtually available memory amount (in bytes)
15969                              <1>       ;           = 4GB - CORE (4MB)
15970                              <1>       ;     ECX = Free memory count (in bytes)
15971                              <1>       ;     EDX = Calculated free memory count (in bytes)
15972                              <1>
15973 0001217D A1[94810100]        <1>       mov   eax, [memory_size] ; in pages
15974 00012182 C1E00C              <1>       shl   eax, 12               ; in bytes
15975 00012185 A3[64030300]        <1>       mov   [u.r0], eax
15976 0001218A E8B421FFFF          <1>       call  calc_free_mem
15977                              <1>       ; edx = calculated free pages
15978                              <1>       ; ecx = 0
15979 0001218F 8B2D[60030300]      <1>       mov   ebp, [u.usp]  ; EBP points to user's registers
15980 00012195 C745100000C0FF      <1>       mov   dword [ebp+16], ECORE ; EBX (for user)
15981                              <1>                             ; 0FFC00000h ; 4GB - 4MB
15982 0001219C C1E20C              <1>       shl   edx, 12
15983 0001219F 895514              <1>       mov   [ebp+20], edx ; EDX (for user)
15984 000121A2 8B0D[98810100]      <1>       mov   ecx, [free_pages]
15985 000121A8 C1E10C              <1>       shl   ecx, 12      ; free bytes
15986 000121AB 894D18              <1>       mov   [ebp+24], ecx ; ECX (for user)
15987                              <1>       ;mov  [free_pages], edx
15988 000121AE E939B7FFFF          <1>       jmp   sysret
15989                              <1>
15990                              <1> sysprompt:
15991                              <1>       ; Set TRDOS 386 Command Interpreter (MainProg) prompt
15992                              <1>       ; 31/12/2017 (TRDOS 386 = TRDOS v2.0)
15993                              <1>       ;
15994                              <1>         ; INPUT ->
15995                              <1>       ;     EBX = 0 -> use default prompt
15996                              <1>       ;     EBX > 0 -> prompt string (ASCIIZ) address
15997                              <1>       ;                (Max. 11 characters except ZERO tail)
15998                              <1>       ; OUTPUT ->
15999                              <1>       ;     (EAX = 0)
16000                              <1>       ;     CF = 0 -> Successful
16001                              <1>       ;     CF = 1 -> Failed
16002                              <1>
16003 000121B3 21DB                <1>       and   ebx, ebx
16004 000121B5 750A                <1>       jnz   short sysprompt_0
16005                              <1>
```

```
16006 000121B7 E8B96CFFFF          <1>       call   default_command_prompt ; '['+'TRDOS'+']'
16007 000121BC E92BB7FFFF          <1>       jmp    sysret
16008                              <1>
16009                              <1> sysprompt_0:
16010 000121C1 31C0                <1>       xor    eax, eax
16011 000121C3 A3[64030300]        <1>       mov    [u.r0], eax
16012 000121C8 89DE                <1>       mov    esi, ebx
16013 000121CA B90C000000          <1>       mov    ecx, 12
16014 000121CF 89E5                <1>       mov    ebp, esp
16015 000121D1 29CC                <1>       sub    esp, ecx
16016 000121D3 49                  <1>       dec    ecx ; 11
16017 000121D4 89E7                <1>       mov    edi, esp
16018 000121D6 E833F1FFFF          <1>       call   transfer_from_user_buffer
16019 000121DB 7211                <1>       jc     short sysprompt_err
16020 000121DD 803E20              <1>       cmp    byte [esi], 20h
16021 000121E0 760C                <1>       jna    short sysprompt_err
16022 000121E2 E8A06CFFFF          <1>       call   set_command_prompt
16023 000121E7 89EC                <1>       mov    esp, ebp
16024 000121E9 E9FEB6FFFF          <1>       jmp    sysret
16025                              <1> sysprompt_err:
16026                              <1> syspath_err:
16027 000121EE 89EC                <1>       mov    esp, ebp
16028 000121F0 E9D7B6FFFF          <1>       jmp    error
16029                              <1>
16030                              <1> syspath:
16031                              <1>       ; Get/Set Run Path
16032                              <1>       ; 31/12/2017 (TRDOS 386 = TRDOS v2.0)
16033                              <1>       ;
16034                              <1>         ; INPUT ->
16035                              <1>       ;      EBX = 0 -> get path (to buffer address in ECX)
16036                              <1>       ;      EBX > 0 -> set path
16037                              <1>       ;          EBX = Path string buffer address (ASCIIZ)
16038                              <1>       ;               (Path description except 'PATH=')
16039                              <1>       ;      ECX = Buffer address (if EBX = 0)
16040                              <1>       ;            (ECX will not be used if EBX > 0)
16041                              <1>       ;      DL = Buffer size (0 = 256 byte)
16042                              <1>       ;
16043                              <1>       ; OUTPUT ->
16044                              <1>       ;      CF = 0 -> Successful (EAX = String length)
16045                              <1>       ;      CF = 1 -> Failed (EAX = 0)
16046                              <1>       ;
16047                              <1>       ; NOTE: 'PATH=' or 'PATH' must be excluded
16048                              <1>       ;  (It must not be at the beginning of the string.)
16049                              <1>
16050 000121F5 89E5                <1>       mov    ebp, esp
16051 000121F7 81EC00010000        <1>       sub    esp, 256
16052 000121FD 89E7                <1>       mov    edi, esp
16053                              <1>
16054 000121FF 31C0                <1>       xor    eax, eax
16055 00012201 A3[64030300]        <1>       mov    [u.r0], eax
16056                              <1>
16057 00012206 21DB                <1>       and    ebx, ebx
16058 00012208 752E                <1>       jnz    short syspath_0
16059                              <1>
16060                              <1>       ; EBX = 0 -> get run path
16061 0001220A 89CB                <1>       mov    ebx, ecx ; buffer addr (in user's mem space)
16062 0001220C BE[EF380100]        <1>       mov    esi, Cmd_Path  ; 'PATH' address
16063 00012211 0FB6CA              <1>       movzx  ecx, dl
16064 00012214 80E901              <1>       sub    cl, 1 ; 0 -> 255, 1 -> 0
16065 00012217 6683D101            <1>       adc    cx, 1 ; 255 -> 256, 0 -> 1
16066                              <1>       ; EDI = Output buffer
16067                              <1>       ; CX = Buffer length
16068                              <1>       ; AL = 0 -> use ASCIIZ word in [ESI]
16069                              <1>       ; ESI = 'PATH' address (with zero tail)
16070 0001221B E89B84FFFF          <1>       call   get_environment_string
16071 00012220 72CC                <1>       jc     short syspath_err
16072 00012222 89DF                <1>       mov    edi, ebx ; User's buffer address
16073 00012224 89E6                <1>       mov    esi, esp
16074                              <1>       ; EDI = User's buffer address
16075                              <1>       ; ECX = transfer (byte) count
16076 00012226 E899F0FFFF          <1>       call   transfer_to_user_buffer
16077 0001222B 72C1                <1>       jc     short syspath_err
16078 0001222D 890D[64030300]      <1>       mov    [u.r0], ecx
16079 00012233 E9B4B6FFFF          <1>       jmp    sysret
16080                              <1>
16081                              <1> syspath_0:
16082 00012238 89DE                <1>       mov    esi, ebx
16083 0001223A 0FB6CA              <1>       movzx  ecx, dl
16084 0001223D 80E901              <1>       sub    cl, 1 ; 0 -> 255, 1 -> 0
16085 00012240 6683D101            <1>       adc    cx, 1 ; 255 -> 256, 0 -> 1
16086 00012244 E8C5F0FFFF          <1>       call   transfer_from_user_buffer
16087 00012249 72A3                <1>       jc     short syspath_err
16088                              <1>       ;(*) 'PATH=' will be added to
16089                              <1>       ;         the head of the string
16090 0001224B 83EC08              <1>       sub    esp, 8 ;(*)
16091 0001224E 89FE                <1>       mov    esi, edi ;(*)
16092 00012250 E84A84FFFF          <1>       call   set_path_x ;(*)
16093 00012255 7297                <1>       jc     short syspath_err
16094 00012257 8915[64030300]      <1>       mov    [u.r0], edx ; run path string length
16095 0001225D E98AB6FFFF          <1>       jmp    sysret
16096                              <1>
16097                              <1> sysenv:
16098                              <1>       ; Get/Set Environment Variables
16099                              <1>       ; 31/12/2017 (TRDOS 386 = TRDOS v2.0)
16100                              <1>       ;
16101                              <1>         ; INPUT ->
16102                              <1>       ;      EBX = 0 -> get (all) environment variables
16103                              <1>       ;            (Required Buffer length = 512 bytes)
16104                              <1>       ;      EBX > 0 -> set (one) environment variable
16105                              <1>       ;            (If there is not a '=' after
16106                              <1>       ;            the environment variable name, it will
16107                              <1>       ;            accepted as 'get environment variable'.)
16108                              <1>       ;             EBX = Buffer address
16109                              <1>       ;      ECX = Buffer address (if EBX = 0)
16110                              <1>       ;            (ECX will not be used if EBX > 0)
```

```
16111                           <1>       ;           (Note: Buffer size is 512 bytes.)
16112                           <1>       ;     DL = Buffer size (0 = 256 byte)
16113                           <1>       ;          (For one envrionment variable)
16114                           <1>       ;
16115                           <1>       ; OUTPUT ->
16116                           <1>       ;     (EAX = 0)
16117                           <1>       ;     CF = 0 -> Successful (EAX = String length)
16118                           <1>       ;     CF = 1 -> Failed (EAX = 0)
16119                           <1>       ;
16120                           <1>       ; Note: Environment variable name, for example,
16121                           <1>       ;     'PATH=' must be included at the beginning
16122                           <1>       ;     of the environment string. If the variable
16123                           <1>       ;     name is as 'PATH' but it is not as 'PATH='
16124                           <1>       ;     the variable string (row) will be returned.
16125                           <1>       ;     If variable name is as 'PATH=' but there is
16126                           <1>       ;     not a following text after the variable name,
16127                           <1>       ;     the environment variable will be reset/deleted.
16128                           <1>
16129 00012262 89E5             <1>       mov   ebp, esp
16130 00012264 81EC00020000     <1>       sub   esp, 512
16131 0001226A 89E7             <1>       mov   edi, esp
16132                           <1>
16133 0001226C 31C0             <1>       xor   eax, eax
16134 0001226E A3[64030300]     <1>       mov   [u.r0], eax
16135                           <1>
16136 00012273 21DB             <1>       and   ebx, ebx
16137 00012275 7524             <1>       jnz   short sysenv_0
16138                           <1>
16139                           <1>       ; EBX = 0 -> get (all) environment variables
16140 00012277 89EC             <1>       mov   esp, ebp
16141 00012279 BE00300900       <1>       mov   esi, Env_Page  ; Environment page
16142 0001227E 89CF             <1>       mov   edi, ecx ; buffer addr (in user's mem space)
16143 00012280 B900020000       <1>       mov   ecx, 512
16144 00012285 E83AF0FFFF       <1>       call  transfer_to_user_buffer
16145 0001228A 0F823CB6FFFF     <1>       jc    error
16146 00012290 890D[64030300]   <1>       mov   [u.r0], ecx
16147 00012296 E951B6FFFF       <1>       jmp   sysret
16148                           <1>
16149                           <1> sysenv_0:
16150 0001229B 89DE             <1>       mov   esi, ebx ; * ; user's buffer address
16151 0001229D 0FB6CA           <1>       movzx ecx, dl
16152 000122A0 80E901           <1>       sub   cl, 1 ; 0 -> 255, 1 -> 0
16153 000122A3 6683D101         <1>       adc   cx, 1 ; 255 -> 256, 0 -> 1
16154 000122A7 E862F0FFFF       <1>       call  transfer_from_user_buffer
16155 000122AC 723F             <1>       jc    short sysenv_err
16156 000122AE 89FE             <1>       mov   esi, edi
16157 000122B0 8A06             <1>       mov   al, [esi]
16158 000122B2 3C20             <1>       cmp   al, 20h
16159 000122B4 7637             <1>       jna   short sysenv_err
16160 000122B6 3C3D             <1>       cmp   al, '='
16161 000122B8 7433             <1>       je    short sysenv_err
16162 000122BA 56               <1>       push  esi
16163                           <1> sysenv_1:
16164 000122BB 46               <1>       inc   esi
16165 000122BC 803E3D           <1>       cmp   byte [esi], '='
16166 000122BF 7433             <1>       je    short sysenv_3
16167 000122C1 803E20           <1>       cmp   byte [esi], 20h
16168 000122C4 73F5             <1>       jnb   short sysenv_1
16169 000122C6 C60600           <1>       mov   byte [esi], 0
16170 000122C9 5E               <1>       pop   esi
16171                           <1>       ; EDI = Output buffer
16172                           <1>       ; CX = Buffer length
16173 000122CA 30C0             <1>       xor   al, al
16174                           <1>       ; AL = 0 -> use ASCIIZ word in [ESI]
16175                           <1>       ; ESI = Environment variable name address
16176 000122CC E8EA83FFFF       <1>       call  get_environment_string
16177 000122D1 721A             <1>       jc    short sysenv_err
16178 000122D3 89DF             <1>       mov   edi, ebx ; * ; user's buffer address
16179 000122D5 89C1             <1>       mov   ecx, eax ; String length
16180 000122D7 89E6             <1>       mov   esi, esp
16181                           <1>       ; ESI = system buffer address
16182                           <1>       ; EDI = User's buffer address
16183                           <1>       ; ECX = transfer (byte) count
16184 000122D9 E8E6EFFFFF       <1>       call  transfer_to_user_buffer
16185 000122DE 720D             <1>       jc    short sysenv_err
16186 000122E0 890D[64030300]   <1>       mov   [u.r0], ecx ; transfer (byte) count
16187                           <1> sysenv_2:
16188 000122E6 89EC             <1>       mov   esp, ebp
16189 000122E8 E9FFB5FFFF       <1>       jmp   sysret
16190                           <1> sysenv_err:
16191 000122ED 89EC             <1>       mov   esp, ebp
16192 000122EF E9D8B5FFFF       <1>       jmp   error
16193                           <1> sysenv_3:
16194 000122F4 46               <1>       inc   esi
16195 000122F5 803E20           <1>       cmp   byte [esi], 20h
16196 000122F8 73FA             <1>       jnb   short sysenv_3
16197 000122FA C60600           <1>       mov   byte [esi], 0
16198 000122FD 5E               <1>       pop   esi
16199 000122FE E87B84FFFF       <1>       call  set_environment_string
16200 00012303 72E8             <1>       jc    short sysenv_err
16201 00012305 8915[64030300]   <1>       mov   [u.r0], edx
16202 0001230B EBD9             <1>       jmp   short sysenv_2
16203                           <1>
16204                           <1>
16205                           <1> ; 22/01/2021
16206                           <1> ; temporary - 24/01/2016
16207                           <1>
16208                           <1> iget:
16209                           <1>       ;retn
16210                           <1> isintr:
16211                           <1>       ;retn
16212                           <1> iopen:
16213                           <1>       ;retn
16214                           <1> iclose:
16215                           <1>       ;retn
```

```
16216                                   <1> sndc:
16217                                   <1>       ;retn
16218                                   <1> access:
16219                                   <1>       ;retn
16220                                   <1> sleep:
16221 0001230D C3                       <1>       retn
 3085                                       %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 0001230E 807E0500                 <1>       cmp    byte [esi+LD_LBAYes], 0
   20 00012312 777B                     <1>         ja      short lba_write
   21                                   <1>
   22                                   <1> chs_write:
   23                                   <1>       ; 25/02/2016
   24                                   <1>       ; 23/02/2016
   25 00012314 C605[458A0100]03         <1>       mov    byte [disk_rw_op], 3 ; CHS write
   26 0001231B 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 0001231D 807E0500                 <1>       cmp    byte [esi+LD_LBAYes], 0
   47 00012321 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>       ;
   57                                   <1>       ; INPUT -> EAX = Logical Block Address
   58                                   <1>       ;         ECX = Number of sectors to read
   59                                   <1>       ;         ESI = Logical Dos Disk Table Offset (DRV)
   60                                   <1>       ;         EBX = Destination Buffer
   61                                   <1>       ; OUTPUT ->
   62                                   <1>       ;         cf = 0 or cf = 1
   63                                   <1>       ; (Modified registers: EAX; EBX, ECX, EDX)
   64                                   <1>
   65                                   <1>       ; 23/02/2016
   66 00012323 C605[458A0100]02         <1>       mov    byte [disk_rw_op], 2 ; CHS read
   67                                   <1>
   68                                   <1> chs_rw:
   69                                   <1>       ;;movzx      edx, word [esi+LD_BPB+SecPerTrack]
   70                                   <1>       ;movzx edx, byte [esi+LD_BPB+SecPerTrack] ; <= 63
   71                                   <1>       ;mov   [disk_rw_spt], dl
   72                                   <1>
   73                                   <1> chs_read_next_sector:
   74 0001232A C605[468A0100]04         <1>       mov    byte [retry_count], 4
   75                                   <1>
   76                                   <1> chs_read_retry:
   77                                   <1>       ;mov   [sector_count], ecx ; 23/02/2016
   78                                   <1>
   79 00012331 50                       <1>       push   eax                ; Linear sector #
   80 00012332 51                       <1>       push   ecx                ; # of FAT/FILE/DIR sectors
   81                                   <1>
   82 00012333 0FB74E1E                 <1>       movzx  ecx, word [esi+LD_BPB+SecPerTrack]
   83                                   <1>       ;movzx ecx, byte [disk_rw_spt] ; 23/02/2016
   84 00012337 29D2                     <1>       sub    edx, edx
   85 00012339 F7F1                     <1>       div    ecx
   86                                   <1>       ; eax = track, dx (dl ) = sector (on track)
   87                                   <1>       ;sub   cl, dl ; 24/02/2016 (spt - sec)
   88                                   <1>       ;push  ecx ; *
   89 0001233B 6689D1                   <1>       mov    cx, dx             ; Sector (zero based)
   90 0001233E 6641                     <1>       inc    cx                 ; To make it 1 based
   91 00012340 6651                     <1>       push   cx
   92 00012342 668B4E20                 <1>       mov    cx, [esi+LD_BPB+Heads]
   93 00012346 6629D2                   <1>       sub    dx, dx
   94 00012349 F7F1                     <1>       div    ecx                ; Convert track to head & cyl
   95                                   <1>       ; eax (ax) = cylinder, dx (dl) = head (max. FFh)
   96 0001234B 88D6                     <1>       mov    dh, dl
   97 0001234D 6659                     <1>       pop    cx                 ; AX=Cyl, DH=Head, CX=Sector
   98 0001234F 8A5602                   <1>       mov    dl, [esi+LD_PhyDrvNo]
```

```
 99                                     <1>
100 00012352 88C5                      <1>       mov    ch, al              ; NOTE: max. 1023 cylinders !
101 00012354 C0CC02                    <1>       ror    ah, 2               ; Rotate 2 bits right
102 00012357 08E1                      <1>       or     cl, ah
103                                    <1>
104                                    <1>       ; 24/02/2016
105                                    <1>       ;pop   eax ; * (spt - sec) (example: 63 - 0 = 63)
106                                    <1>       ;cmp   eax, [sector_count]
107                                    <1>       ;jb    short chs_write_sectors
108                                    <1>       ;je    short chs_read_sectors
109                                    <1>       ;; (# of sectors to read is more than remaining sectors on the track)
110                                    <1>       ;mov   al, [sector_count]
111                                    <1> ;chs_read_sectors: ; read or write !
112 00012359 B001                      <1>       mov    al, 1 ; 25/02/2016
113 0001235B 8A25[458A0100]            <1>       mov    ah, [disk_rw_op]  ; 02h = chs read, 03h = chs write
114                                    <1>       ;
115 00012361 E8B92EFFFF                <1>       call   int13h              ; BIOS Service func ( ah ) = 2
116                                    <1>                                  ; Read disk sectors
117                                    <1>                                  ; AL-sec num CH-track CL-sec
118                                    <1>                                  ; DH-head DL-drive ES:BX-buffer
119                                    <1>                                  ; CF-flag AH-stat AL-sec read
120                                    <1>                                ; If CF = 1 then (If AH > 0)
121 00012366 8825[478A0100]            <1>       mov    [disk_rw_err], ah
122                                    <1>
123 0001236C 59                        <1>       pop    ecx
124 0001236D 58                        <1>       pop    eax
125 0001236E 7314                      <1>       jnc    short chs_read_ok
126                                    <1>
127 00012370 803D[478A0100]09          <1>       cmp    byte [disk_rw_err], 09h ; DMA crossed 64K segment boundary
128 00012377 7408                      <1>       je     short chs_read_error_retn
129                                    <1>
130 00012379 FE0D[468A0100]            <1>       dec    byte [retry_count]
131 0001237F 75B0                      <1>       jnz    short chs_read_retry
132                                    <1>
133                                    <1> chs_read_error_retn:
134 00012381 F9                        <1>       stc
135                                    <1>       ;retn
136 00012382 EB69                      <1>       jmp    short update_drv_error_byte
137                                    <1>
138                                    <1> ;chs_write_sectors: ; read or write
139                                    <1>       ;; (# of sectors to read is less than remaining sectors on the track)
140                                    <1>       ;mov   [sector_count], al
141                                    <1>       ;jmp   short chs_read_sectors
142                                    <1>
143                                    <1> chs_read_ok:
144                                    <1>       ;; 23/02/2016
145                                    <1>       ;movzx edx, byte [sector_count] ; sector count (<= spt)
146                                    <1>         ;sub    ecx, edx  ; remaining sector count
147                                    <1>       ;jna   short update_drv_error_byte
148                                    <1>       ;add   eax, edx ; next disk sector
149                                    <1>       ;shl   edx, 9 ; 512 * sector count
150                                    <1>       ;add   ebx, edx ; next buffer byte address
151                                    <1>         ;jmp     chs_read_next_sector
152                                    <1>       ; 25/02/2016
153 00012384 40                        <1>       inc    eax ; next sector
154 00012385 81C300020000              <1>       add    ebx, 512
155 0001238B E29D                      <1>       loop   chs_read_next_sector
156 0001238D EB5E                      <1>       jmp    short update_drv_error_byte
157                                    <1>
158                                    <1> lba_write:
159                                    <1>       ; 23/02/2016
160 0001238F C605[458A0100]1C          <1>       mov    byte [disk_rw_op], 1Ch ; LBA write
161 00012396 EB07                      <1>       jmp    short lba_rw
162                                    <1>
163                                    <1> lba_read:
164                                    <1>       ; 23/02/2016
165                                    <1>       ; 17/02/2016
166                                    <1>       ; 14/02/2016
167                                    <1>       ; 13/02/2016
168                                    <1>       ; 31/01/2016 (TRDOS 386 =  TRDOS v2.0)
169                                    <1>       ; 10/07/2015 (Retro UNIX 386 v1)
170                                    <1>       ;
171                                    <1>       ; INPUT -> EAX = Logical Block Address
172                                    <1>       ;         ESI = Logical Dos Disk Table Offset (DRV)
173                                    <1>       ;         ECX = Sector Count
174                                    <1>       ;         EBX = Destination Buffer
175                                    <1>       ; OUTPUT ->
176                                    <1>       ;         cf = 0 or cf = 1
177                                    <1>       ; (Modified registers: EAX, EBX, ECX, EDX)
178                                    <1>
179                                    <1>       ; LBA read/write (with private LBA function)
180                                    <1>       ;((Retro UNIX 386 v1 - DISK I/O code by Erdogan Tan))
181                                    <1>
182                                    <1>
183                                    <1>       ; 23/02/2016
184 00012398 C605[458A0100]1B          <1>       mov    byte [disk_rw_op], 1Bh ; LBA read
185                                    <1>
186                                    <1> lba_rw:
187                                    <1>       ; 17/02/2016
188 0001239F 57                        <1>       push   edi
189                                    <1>
190 000123A0 890D[488A0100]            <1>       mov    [sector_count], ecx ; total sector (read) count
191                                    <1>
192 000123A6 8A5602                    <1>       mov    dl, [esi+LD_PhyDrvNo]
193                                    <1>       ; dl = physical drive number (0,1, 80h, 81h, 82h, 83h)
194                                    <1>
195                                    <1> lba_read_next:
196 000123A9 81F900010000              <1>       cmp    ecx, 256
197 000123AF 7605                      <1>       jna    short lba_read_rsc
198 000123B1 B900010000                <1>       mov    ecx, 256 ; 17/02/2016
199                                    <1> lba_read_rsc:
200 000123B6 290D[488A0100]            <1>       sub    [sector_count], ecx ; remain sectors
201                                    <1>
202 000123BC 89CF                      <1>       mov    edi, ecx
203 000123BE 89C1                      <1>       mov    ecx, eax ; sector number/address
```

```
                                <1>
000123C0 C605[468A0100]04       <1>       mov   byte [retry_count], 4
                                <1> lba_read_retry:
000123C7 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
000123C9 8A25[458A0100]         <1>       mov   ah, [disk_rw_op] ; 1Bh = LBA read, 1Ch = LBA write
000123CF E84B2EFFFF             <1>       call  int13h
                                <1>       ; al = ? (changed)
                                <1>       ; ah = error code
000123D4 8825[478A0100]         <1>       mov   [disk_rw_err], ah
000123DA 7334                   <1>       jnc   short lba_read_ok
000123DC 80FC80                 <1>       cmp   ah, 80h ; time out?
000123DF 740A                   <1>         je      short lba_read_stc_retn
000123E1 FE0D[468A0100]         <1>       dec   byte [retry_count]
000123E7 7FDE                   <1>       jg    short lba_read_retry
000123E9 743A                   <1>       jz    short lba_read_reset
                                <1>       ; sf =  1
                                <1>
                                <1> lba_read_stc_retn:
000123EB F9                     <1>       stc
                                <1> lba_read_retn:
000123EC 5F                     <1>       pop   edi
                                <1>
                                <1> update_drv_error_byte:
000123ED 9C                     <1>       pushf
000123EE 53                     <1>       push  ebx
000123EF 6651                   <1>       push  cx
                                <1>       ;or   ecx, ecx
                                <1>       ;jz   short udrv_errb0
000123F1 8A0D[478A0100]         <1>       mov   cl, [disk_rw_err]
                                <1> udrv_errb0:
000123F7 0FB65E02               <1>       movzx ebx, byte [esi+LD_PhyDrvNo]
000123FB 80FB02                 <1>       cmp   bl, 2
000123FE 7203                   <1>         jb      short udrv_errb1
00012400 80EB7E                 <1>       sub   bl, 7Eh
                                <1>       ;cmp  bl, 5
                                <1>       ;ja   short udrv_errb2
                                <1> udrv_errb1:
00012403 81C3[416E0000]         <1>         add     ebx, drv.error ; 13/02/2016
00012409 880B                   <1>       mov   [ebx], cl ; error code
                                <1> udrv_errb2:
0001240B 6659                   <1>       pop   cx
0001240D 5B                     <1>       pop   ebx
0001240E 9D                     <1>       popf
0001240F C3                     <1>       retn
                                <1>
                                <1> lba_read_ok:
00012410 89C8                   <1>       mov   eax, ecx ; sector number
00012412 01F8                   <1>       add   eax, edi ; sector number (next)
00012414 C1E709                 <1>       shl   edi, 9 ; sector count * 512
00012417 01FB                   <1>       add   ebx, edi ; next buffer offset
                                <1>
00012419 8B0D[488A0100]         <1>       mov   ecx, [sector_count] ; remaining sectors
0001241F 09C9                   <1>       or    ecx, ecx
00012421 7586                   <1>       jnz   short lba_read_next
00012423 EBC7                   <1>       jmp   short lba_read_retn
                                <1>
                                <1> lba_read_reset:
00012425 B40D                   <1>       mov   ah, 0Dh ; Alternate reset
00012427 E8F32DFFFF             <1>         call int13h
                                <1>       ; al = ? (changed)
                                <1>       ; ah = error code
0001242C 7399                   <1>       jnc   short lba_read_retry
0001242E 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: 03/08/2020
                                <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:
00012430 3A05[B3030300]                 <1>         cmp     al, [u.uno]     ; same process ?
00012436 750C                           <1>       jne   short srunseq_2 ; no
                                        <1>
00012438 8A25[A9030300]                 <1>       mov   ah, [u.pri]  ; present/current priority
0001243E 80FC02                         <1>       cmp   ah, 2        ; 'run for event' priority level
00012441 7221                           <1>       jb    short srunseq_6 ; no
                                        <1>
                                        <1> srunseq_1:
                                        <1>       ; there is nothing to do!
00012443 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
00012444 80FA02                         <1>       cmp   dl, 2        ; event queue
00012447 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
00012449 3A15[A9030300]                 <1>       cmp   dl, [u.pri]  ; priority of present process
```

```
0001244F 7606                <1>       jna    short srunseq_3 ; is high, also
                             <1>       ;
                             <1>       ; present process will be swapped/switched out
00012451 FE05[218E0100]      <1>       inc    byte [p_change] ; 1
                             <1>
                             <1> srunseq_3:
                             <1>       ; add process to 'runq_event' queue for new event
00012457 BB[52030300]        <1>       mov    ebx, runq_event ; high priority run queue
                             <1>
                             <1> srunseq_4:
                             <1>       ; al = process number
                             <1>       ; ebx = run queue
0001245C E887EDFFFF          <1>       call   putlu
00012461 C3                  <1>       retn
                             <1>
                             <1> srunseq_5:
00012462 F5                  <1>       cmc
00012463 C3                  <1>       retn
                             <1>
                             <1> srunseq_6:
                             <1>       ; present priority of the process is not high
                             <1>
00012464 8815[A9030300]      <1>       mov    [u.pri], dl ; new priority
                             <1>                          ; (will be used by 'tswap')
                             <1>
0001246A 80FA02              <1>       cmp    dl, 2        ; high priority ?
0001246D 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>
0001246F BB[54030300]        <1>       mov    ebx, runq_normal ; 'run as regular' queue
                             <1>
00012474 20E4                <1>       and    ah, ah  ;  previous value of [u.pri]
00012476 75E4                <1>       jnz    short srunseq_4
                             <1>
00012478 43                  <1>       inc    ebx
00012479 43                  <1>       inc    ebx
                             <1>       ; ebx = runq_background ; 'run on backgroud' queue
                             <1>
0001247A 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>
0001247C 803D[A8030300]00    <1>       cmp    byte [u.quant], 0
00012483 772C                <1>       ja     short clk_1
                             <1>       ;
00012485 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.
0001248C 7623                <1>       jna    short clk_1 ; yes, do not swap out
                             <1>       ;
0001248E 803D[5B030300]FF    <1>       cmp     byte [sysflg], 0FFh ; user or system space ?
00012495 7520                <1>       jne    short clk_2      ; system space (sysflg <> 0FFh)
                             <1>       ;
00012497 66833D[AA030300]00  <1>       cmp    word [u.intr], 0
0001249F 7616                <1>       jna    short clk_2
                             <1>       ;
                             <1>       ; 23/05/2016
000124A1 803D[228E0100]00    <1>       cmp    byte [multi_tasking], 0
000124A8 760D                <1>       jna    short clk_2
                             <1>       ;
000124AA FE05[218E0100]      <1>       inc    byte [p_change] ; it is time to change running process
000124B0 C3                  <1>       retn
                             <1> clk_1:
000124B1 FE0D[A8030300]      <1>       dec    byte [u.quant]
                             <1> clk_2:
000124B7 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
000124B8 80FC07         <1>       cmp    ah, 7
000124BB 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
000124BD FB             <1>       sti
                        <1>
                        <1>       ;sti   ; enable interrupts
000124BE 80642408FE     <1>       and    byte [esp+8], 11111110b   ; clear carry bit of eflags register
                        <1>
000124C3 80FC01         <1>       cmp    ah, 1
000124C6 7205           <1>       jb     short int34h_0
000124C8 7705           <1>       ja     short int34h_1
                        <1>
000124CA EE             <1>       out    dx, al
                        <1>       ;iretd
000124CB EB01           <1>       jmp short int34h_iret
                        <1>
                        <1> int34h_0:
000124CD 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
000124CE CF             <1>       iretd
                        <1>
                        <1> int34h_1:
000124CF F6C401         <1>       test   ah, 1
000124D2 7516           <1>       jnz    short int34h_3 ; out
                        <1>
                        <1>       ; in
000124D4 80FC02         <1>       cmp    ah, 2
000124D7 7707           <1>       ja     short int34h_2
                        <1>
000124D9 6689D8         <1>       mov    ax, bx
000124DC 66ED           <1>       in     ax, dx
                        <1>       ;iretd
000124DE EBEE           <1>       jmp    short int34h_iret
                        <1>
                        <1> int34h_2:
000124E0 80FC04         <1>       cmp    ah, 4
000124E3 772C           <1>       ja     short int34h_7      ; 12/10/2017
                        <1>       ; ah = 4
000124E5 89D8           <1>       mov    eax, ebx
000124E7 ED             <1>       in     eax, dx
                        <1>       ;iretd
000124E8 EBE4           <1>       jmp    short int34h_iret
                        <1>
                        <1> int34h_3:
000124EA 80FC03         <1>       cmp    ah, 3
000124ED 7707           <1>       ja     short int34h_4
                        <1>
000124EF 6689D8         <1>       mov    ax, bx
000124F2 66EF           <1>       out    dx, ax
                        <1>       ;iretd
000124F4 EBD8           <1>       jmp    short int34h_iret
                        <1>
                        <1> int34h_4:
000124F6 80FC05         <1>       cmp    ah, 5
000124F9 770B           <1>       ja     short int34h_6      ; 12/10/2017
                        <1>       ; ah = 5
000124FB 89D8           <1>       mov    eax, ebx
000124FD EF             <1>       out    dx, eax
                        <1>       ;iretd
000124FE EBCE           <1>       jmp    short int34h_iret
                        <1>
                        <1> int34h_5:
00012500 804C240801     <1>       or     byte [esp+8], 1     ; set carry bit of eflags register
00012505 CF             <1>       iretd
                        <1>
                        <1>       ; 12/10/2017
                        <1> int34h_6:
00012506 6689D8         <1>       mov    ax, bx
00012509 EE             <1>       out    dx, al
0001250A EB00           <1>       jmp    short $+2
0001250C 86E0           <1>       xchg   ah, al
0001250E EE             <1>       out    dx, al
                        <1>       ;xchg  al, ah
                        <1>       ;iretd
0001250F EB06           <1>       jmp    short int34h_8
                        <1> int34h_7:
00012511 EC             <1>       in     al, dx
00012512 EB00           <1>       jmp    short $+2
00012514 88C4           <1>       mov    ah, al
00012516 EC             <1>       in     al, dx
                        <1> int34h_8:
00012517 86C4           <1>       xchg   al, ah
```

```
00012519 CF                 <1>       iretd
                            <1>
                            <1>
                            <1> INT4Ah:
                            <1>       ; 24/01/2016
                            <1>       ; this procedure will be called by 'RTC_INT' (in 'timer.s')
0001251A 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>       ;
0001251B 890424             <1>       mov    [esp], eax ; overwrite call return address
                            <1>       ;push  eax
0001251E 66B80900           <1>       mov    ax, 9
00012522 EB07               <1>       jmp    short comm_int
                            <1> com1_int:
                            <1>       ; 07/11/2015
                            <1>       ; 24/10/2015
00012524 890424             <1>       mov    [esp], eax ; overwrite call return address
                            <1>       ; 23/10/2015
                            <1>       ;push  eax
00012527 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
0001252B 53                 <1>       push   ebx
0001252C 56                 <1>       push   esi
0001252D 57                 <1>       push   edi
0001252E 1E                 <1>       push   ds
0001252F 06                 <1>       push   es
                            <1>       ; 18/11/2015
00012530 0F20DB             <1>       mov    ebx, cr3
00012533 53                 <1>       push   ebx ; ****
                            <1>       ;
00012534 51                 <1>       push   ecx ; ***
00012535 52                 <1>       push   edx ; **
                            <1>       ;
00012536 BB10000000         <1>       mov    ebx, KDATA
0001253B 8EDB               <1>       mov    ds, bx
0001253D 8EC3               <1>       mov    es, bx
                            <1>       ;
0001253F 8B0D[90810100]     <1>       mov    ecx, [k_page_dir]
00012545 0F22D9             <1>       mov    cr3, ecx
                            <1>       ; 20/11/2015
                            <1>       ; Interrupt identification register
00012548 66BAFA02           <1>       mov    dx, 2FAh ; COM2
                            <1>       ;
0001254C 3C08               <1>       cmp    al, 8
0001254E 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>       ;
00012550 FEC6               <1>       inc    dh ; 3FAh ; COM1 Interrupt id. register
                            <1> com_i0:
                            <1>       ;push  eax ; *
                            <1>       ; 07/11/2015
00012552 A2[FA810100]       <1>       mov    byte [ccomport], al
                            <1>       ; 09/11/2015
00012557 0FB7D8             <1>       movzx  ebx, ax ; 8 or 9
                            <1>       ; 17/11/2015
                            <1>       ; reset request for response status
0001255A 88A3[F0810100]     <1>       mov    [ebx+req_resp-8], ah ; 0
                            <1>       ;
                            <1>       ; 20/11/2015
00012560 EC                 <1>       in     al, dx       ; read interrupt id. register
00012561 EB00               <1>       JMP    $+2          ; I/O DELAY
00012563 2404               <1>       and    al, 4        ; received data available?
00012565 7470               <1>       jz     short com_eoi; (transmit. holding reg. empty)
                            <1>       ;
                            <1>       ; 20/11/2015
00012567 80EA02             <1>       sub    dl, 3FAh-3F8h; data register (3F8h, 2F8h)
0001256A EC                 <1>       in     al, dx       ; read character
                            <1>       ;JMP   $+2          ; I/O DELAY
                            <1>       ; 08/11/2015
                            <1>       ; 07/11/2015
0001256B 89DE               <1>       mov    esi, ebx
0001256D 89DF               <1>       mov    edi, ebx
0001256F 81C6[F4810100]     <1>       add    esi, rchar - 8 ; points to last received char
00012575 81C7[F6810100]     <1>       add    edi, schar - 8 ; points to last sent char
0001257B 8806               <1>       mov    [esi], al ; received char (current char)
```

```
                                <1>       ; query
000125 7D 20C0                  <1>       and    al, al
000125 7F 7527                  <1>       jnz    short com_i2
                                <1>       ; response
                                <1>       ; 17/11/2015
                                <1>       ; set request for response status
00012581 FE83[F0810100]         <1>         inc     byte [ebx+req_resp-8] ; 1
                                <1>       ;
00012587 6683C205               <1>       add    dx, 3FDh-3F8h; (3FDh, 2FDh)
0001258B EC                     <1>       in     al, dx       ; read line status register
0001258C EB00                   <1>       JMP    $+2          ; I/O DELAY
0001258E 2420                   <1>       and    al, 20h             ; transmitter holding reg. empty?
00012590 7445                   <1>       jz     short com_eoi       ; no
00012592 B0FF                   <1>       mov    al, 0FFh     ; response
00012594 6683EA05               <1>       sub    dx, 3FDh-3F8h       ; data port (3F8h, 2F8h)
00012598 EE                     <1>       out    dx, al       ; send on serial port
                                <1>       ; 17/11/2015
00012599 803F00                 <1>       cmp    byte [edi], 0   ; query ? (schar)
0001259C 7502                   <1>       jne    short com_i1    ; no
0001259E 8807                   <1>       mov    [edi], al    ; 0FFh (responded)
                                <1> com_i1:
                                <1>       ; 17/11/2015
                                <1>       ; reset request for response status (again)
000125A0 FE8B[F0810100]         <1>         dec     byte [ebx+req_resp-8] ; 0
000125A6 EB2F                   <1>       jmp    short com_eoi
                                <1> com_i2:
                                <1>       ; 08/11/2015
000125A8 3CFF                   <1>       cmp    al, 0FFh     ; (response ?)
000125AA 7417                   <1>       je     short com_i3 ; (check for response signal)
                                <1>       ; 07/11/2015
000125AC 3C04                   <1>       cmp    al, 04h      ; EOT
000125AE 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)
000125B0 861D[BE810100]         <1>       xchg   bl, [ptty]  ; tty number (8 or 9)
000125B6 E81550FFFF             <1>       call   ctrlbrk
000125BB 861D[BE810100]         <1>       xchg   [ptty], bl ; (restore ptty value and BL value)
                                <1>       ;mov   al, 04h ; EOT
                                <1>       ; 08/11/2015
000125C1 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
000125C3 803F00                 <1>         cmp     byte [edi], 0 ; query ? (schar)
000125C6 7704                   <1>       ja     short com_i4 ; no
                                <1>       ; reset query status (schar)
000125C8 8807                   <1>       mov    [edi], al ; 0FFh
000125CA FEC0                   <1>       inc    al ; 0
                                <1> com_i4:
                                <1>       ; 27/07/2014
                                <1>       ; 09/07/2014
000125CC D0E3                   <1>       shl    bl, 1
000125CE 81C3[C0810100]         <1>       add    ebx, ttychr
                                <1>       ; 23/07/2014 (always overwrite)
                                <1>       ;;cmp  word [ebx], 0
                                <1>       ;;ja   short com_eoi
                                <1>       ;
000125D4 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 ; *
000125D7 A0[FA810100]           <1>       mov    al, byte [ccomport] ; current COM port
                                <1>        ; al = tty number (8 or 9)
000125DC E85E010000             <1>         call wakeup
                                <1> com_iret:
                                <1>       ; 23/10/2015
000125E1 5A                     <1>       pop    edx ; **
000125E2 59                     <1>       pop    ecx ; ***
                                <1>       ; 18/11/2015
                                <1>       ;pop   eax ; ****
                                <1>       ;mov   cr3, eax
                                <1>       ;jmp   iiret
000125E3 E9C3E7FEFF             <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>       ;
000125E8 BB[F6810100]   <1>       mov    ebx, com1p          ; COM1 parameters
000125ED 66BAF803       <1>       mov    dx, 3F8h            ; COM1
                        <1>        ; 29/10/2015
000125F1 66B90103       <1>       mov    cx, 301h  ; divisor = 1 (115200 baud)
000125F5 E86F000000     <1>       call   sp_i3  ; call A4
000125FA A880           <1>       test   al, 80h
000125FC 7410           <1>       jz     short sp_i0 ; OK..
                        <1>              ; Error !
                        <1>       ;mov   dx, 3F8h
000125FE 80EA05         <1>       sub    dl, 5 ; 3FDh -> 3F8h
00012601 66B90E03       <1>       mov    cx, 30Eh  ; divisor = 12 (9600 baud)
00012605 E85F000000     <1>       call   sp_i3  ; call A4
0001260A A880           <1>       test   al, 80h
0001260C 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>       ;
0001260E C603E3         <1>       mov    byte [ebx], 0E3h ; 11100011b
00012611 E8DC000000     <1>       call   sp_i5 ; 29/06/2015
                        <1> sp_i1:
00012616 43             <1>       inc    ebx
00012617 66BAF802       <1>       mov    dx, 2F8h            ; COM2
                        <1>        ; 29/10/2015
0001261B 66B90103       <1>       mov    cx, 301h  ; divisor = 1 (115200 baud)
0001261F E845000000     <1>       call   sp_i3  ; call A4
00012624 A880           <1>       test   al, 80h
00012626 7410           <1>       jz     short sp_i2 ; OK..
                        <1>              ; Error !
                        <1>       ;mov   dx, 2F8h
00012628 80EA05         <1>       sub    dl, 5 ; 2FDh -> 2F8h
0001262B 66B90E03       <1>       mov    cx, 30Eh  ; divisor = 12 (9600 baud)
0001262F E835000000     <1>       call   sp_i3  ; call A4
00012634 A880           <1>       test   al, 80h
00012636 7530           <1>       jnz    short sp_i7
                        <1> sp_i2:
00012638 C603E3         <1>       mov    byte [ebx], 0E3h ; 11100011b
                        <1> sp_i6:
                        <1>       ;; COM2 - enabling IRQ 3
                        <1>       ; 07/11/2015
                        <1>       ; 26/10/2015
0001263B 9C             <1>       pushf
0001263C FA             <1>       cli
                        <1>       ;
0001263D 66BAFC02       <1>       mov    dx, 2FCh            ; modem control register
00012641 EC             <1>       in     al, dx                    ; read register
00012642 EB00           <1>       JMP    $+2                 ; I/O DELAY
00012644 0C08           <1>       or     al, 8               ; enable bit 3 (OUT2)
00012646 EE             <1>       out    dx, al              ; write back to register
00012647 EB00           <1>       JMP    $+2                 ; I/O DELAY
00012649 66BAF902       <1>       mov    dx, 2F9h            ; interrupt enable register
0001264D EC             <1>       in     al, dx              ; read register
0001264E EB00           <1>       JMP    $+2                 ; I/O DELAY
                        <1>       ;or    al, 1               ; receiver data interrupt enable and
00012650 0C03           <1>       or     al, 3               ; transmitter empty interrupt enable
00012652 EE             <1>       out    dx, al                    ; write back to register
00012653 EB00           <1>       JMP    $+2                 ; I/O DELAY
00012655 E421           <1>       in     al, 21h             ; read interrupt mask register
00012657 EB00           <1>       JMP    $+2                 ; I/O DELAY
00012659 24F7           <1>       and    al, 0F7h            ; enable IRQ 3 (COM2)
0001265B E621           <1>       out    21h, al             ; write back to register
                        <1>       ;
                        <1>       ; 23/10/2015
0001265D B8[1B250100]   <1>       mov    eax, com2_int
00012662 A3[3A270100]   <1>       mov    [com2_irq3], eax
```

```
                        <1>       ; 26/10/2015
00012667 9D             <1>       popf
                        <1> sp_i7:
00012668 C3             <1>       retn
                        <1>
                        <1> sp_i3:
                        <1> ;A4:         ;----- INITIALIZE THE COMMUNICATIONS PORT
                        <1>       ; 28/10/2015
00012669 FEC2           <1>       inc    dl     ; 3F9h (2F9h); 3F9h, COM1 Interrupt enable register
0001266B B000           <1>       mov    al, 0
0001266D EE             <1>       out    dx, al              ; disable serial port interrupt
0001266E EB00           <1>       JMP    $+2                 ; I/O DELAY
00012670 80C202         <1>       add    dl, 2  ; 3FBh (2FBh); COM1 Line control register (3FBh)
00012673 B080           <1>       mov    al, 80h
00012675 EE             <1>       out    dx, al              ; SET DLAB=1 ; divisor latch access bit
                        <1>       ;----- SET BAUD RATE DIVISOR
                        <1>       ; 26/10/2015
00012676 80EA03         <1>       sub    dl, 3   ; 3F8h (2F8h)     ; register for least significant byte
                        <1>                                  ; of the divisor value
00012679 88C8           <1>       mov    al, cl ; 1
0001267B 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)
0001267C EB00           <1>       JMP    $+2                 ; I/O DELAY
0001267E 28C0           <1>       sub    al, al
00012680 FEC2           <1>       inc    dl      ; 3F9h (2F9h)     ; register for most significant byte
                        <1>                                  ; of the divisor value
00012682 EE             <1>       out    dx, al ; 0
00012683 EB00           <1>       JMP    $+2                 ; I/O DELAY
                        <1>       ;
00012685 88E8           <1>       mov    al, ch ; 3          ; 8 data bits, 1 stop bit, no parity
                        <1>       ;and   al, 1Fh ; Bits 0,1,2,3,4
00012687 80C202         <1>       add    dl, 2  ; 3FBh (2FBh); Line control register
0001268A EE             <1>       out    dx, al
0001268B EB00           <1>       JMP    $+2                 ; I/O DELAY
                        <1>       ; 29/10/2015
0001268D FECA           <1>       dec    dl     ; 3FAh (2FAh); FIFO Control register (16550/16750)
0001268F 30C0           <1>       xor    al, al              ; 0
00012691 EE             <1>       out    dx, al              ; Disable FIFOs (reset to 8250 mode)
00012692 EB00           <1>       JMP    $+2
                        <1> sp_i4:
                        <1> ;A18: ;----- COMM PORT STATUS ROUTINE
                        <1>       ; 29/06/2015 (line status after modem status)
00012694 80C204         <1>       add    dl, 4  ; 3FEh (2FEh); Modem status register
                        <1> sp_i4s:
00012697 EC             <1>       in     al, dx              ; GET MODEM CONTROL STATUS
00012698 EB00           <1>       JMP    $+2                 ; I/O DELAY
0001269A 88C4           <1>       mov    ah, al              ; PUT IN (AH) FOR RETURN
0001269C FECA           <1>       dec    dl     ; 3FDh (2FDh); POINT TO LINE STATUS REGISTER
                        <1>                                  ; dx = 3FDh for COM1, 2FDh for COM2
0001269E EC             <1>       in     al, dx              ; GET LINE CONTROL STATUS
                        <1>       ; AL = Line status, AH = Modem status
0001269F C3             <1>       retn
                        <1>
                        <1> sp_status:
                        <1>       ; 29/06/2015
                        <1>       ; 27/06/2015 (Retro UNIX 386 v1)
                        <1>       ; Get serial port status
000126A0 66BAFE03       <1>       mov    dx, 3FEh            ; Modem status register (COM1)
000126A4 28C6           <1>       sub    dh, al              ; dh = 2 for COM2 (al = 1)
                        <1>                                  ; dx = 2FEh for COM2
000126A6 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>       ;
000126A8 66BAF803               <1>       mov    dx, 3F8h
000126AC BB[F6810100]           <1>       mov    ebx, com1p ; COM1 control byte offset
000126B1 3C01                   <1>       cmp    al, 1
000126B3 776B                   <1>       ja     short sp_invp_err
000126B5 7203                   <1>       jb     short sp_setp1 ;  COM1 (AL = 0)
000126B7 FECE                   <1>       dec    dh ; 2F8h
000126B9 43                     <1>       inc    ebx ; COM2 control byte offset
                                <1> sp_setp1:
                                <1>       ; 29/10/2015
000126BA 8823                   <1>       mov    [ebx], ah
000126BC 0FB6CC                 <1>       movzx  ecx, ah
000126BF C0E905                 <1>       shr    cl, 5 ; -> baud rate index
000126C2 80E41F                 <1>       and    ah, 1Fh ; communication parameters except baud rate
000126C5 8A81[2F270100]         <1>       mov    al, [ecx+b_div_tbl]
000126CB 6689C1                 <1>       mov    cx, ax
000126CE E896FFFFFF             <1>       call   sp_i3
000126D3 6689C1                 <1>       mov    cx, ax ; CL = Line status, CH = Modem status
000126D6 A880                   <1>       test   al, 80h
000126D8 740F                   <1>       jz     short sp_setp2
000126DA C603E3                 <1>         mov     byte [ebx], 0E3h ; Reset to initial value (11100011b)
                                <1> stp_dnr_err:
000126DD C705[C8030300]0F00-    <1>       mov    dword [u.error], ERR_DEV_NOT_RDY ; 'device not ready !'
000126E5 0000                   <1>
                                <1>       ; CL = Line status, CH = Modem status
000126E7 F9                     <1>       stc
000126E8 C3                     <1>       retn
                                <1> sp_setp2:
000126E9 80FE02                 <1>       cmp    dh, 2 ; COM2 (2F?h)
000126EC 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
000126F2 9C                     <1>       pushf
000126F3 FA                     <1>       cli
000126F4 66BAFC03               <1>       mov    dx, 3FCh            ; modem control register
000126F8 EC                     <1>       in     al, dx                    ; read register
000126F9 EB00                   <1>       JMP    $+2                 ; I/O DELAY
000126FB 0C08                   <1>       or     al, 8               ; enable bit 3 (OUT2)
000126FD EE                     <1>       out    dx, al              ; write back to register
000126FE EB00                   <1>       JMP    $+2                 ; I/O DELAY
00012700 66BAF903               <1>       mov    dx, 3F9h            ; interrupt enable register
00012704 EC                     <1>       in     al, dx              ; read register
00012705 EB00                   <1>       JMP    $+2                 ; I/O DELAY
                                <1>       ;or    al, 1               ; receiver data interrupt enable and
00012707 0C03                   <1>       or     al, 3               ; transmitter empty interrupt enable
00012709 EE                     <1>       out    dx, al                    ; write back to register
0001270A EB00                   <1>       JMP    $+2                 ; I/O DELAY
0001270C E421                   <1>       in     al, 21h             ; read interrupt mask register
0001270E EB00                   <1>       JMP    $+2                 ; I/O DELAY
00012710 24EF                   <1>       and    al, 0EFh            ; enable IRQ 4 (COM1)
00012712 E621                   <1>       out    21h, al             ; write back to register
                                <1>       ;
                                <1>       ; 23/10/2015
00012714 B8[24250100]           <1>       mov    eax, com1_int
00012719 A3[36270100]           <1>       mov    [com1_irq4], eax
                                <1>       ; 26/10/2015
0001271E 9D                     <1>       popf
0001271F C3                     <1>       retn
                                <1>
                                <1> sp_invp_err:
00012720 C705[C8030300]1700-    <1>       mov    dword [u.error], ERR_INV_PARAMETER ; 'invalid parameter !'
00012728 0000                   <1>
0001272A 31C9                   <1>       xor    ecx, ecx
0001272C 49                     <1>       dec    ecx ; 0FFFFh
0001272D F9                     <1>       stc
0001272E C3                     <1>       retn
                                <1>
                                <1> ; 29/10/2015
                                <1> b_div_tbl: ; Baud rate divisor table (115200/divisor)
0001272F 010C0603080401         <1>       db 1, 12, 6, 3, 8, 4, 1
                                <1>
                                <1>
                                <1> ; 23/10/2015
                                <1> com1_irq4:
00012736 [3E270100]             <1>       dd dummy_retn
                                <1> com2_irq3:
0001273A [3E270100]             <1>       dd dummy_retn
                                <1>
                                <1> dummy_retn:
0001273E C3                     <1>       retn
                                <1>
                                <1> wakeup:
                                <1>       ; 24/01/2016
0001273F C3                     <1>       retn
                                <1>
                                <1> set_working_path_x:
                                <1>             ; 17/10/2016 (TRDOS 386 - FFF & FNF)
00012740 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>
00012744 8825[448E0100]         <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>
0001274A 66A3[488E0100] <1>             mov    [SWP_Mode], ax
00012750 A0[56820100]   <1>             mov    al, [Current_Drv]
00012755 30E4           <1>             xor    ah, ah
00012757 66A3[4A8E0100] <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>
0001275D 55             <1>             push   ebp
0001275E 89E5           <1>             mov    ebp, esp
                        <1>
00012760 B980000000     <1>             mov    ecx, 128 ; maximum path length = 128 bytes
00012765 29CC           <1>             sub    esp, ecx ; reserve 128 bytes (buffer) on stack
00012767 89E7           <1>             mov    edi, esp ; destination address (kernel space)
                        <1>             ; esi = source address (virtual, in user's memory space)
00012769 E8A0EBFFFF     <1>             call   transfer_from_user_buffer
0001276E 720A           <1>             jc     short loc_swp_xor_retn
                        <1>
00012770 89E6           <1>             mov    esi, esp ; temporary buffer (the path) on stack
                        <1> loc_swp_fchar:
00012772 8A06           <1>             mov    al, [esi]
00012774 3C20           <1>             cmp    al, 20h
00012776 7711           <1>             ja     short loc_swp_parse_path_name
00012778 740C           <1>             je     short loc_swp_fchar_next
                        <1>
                        <1> loc_swp_xor_retn:
0001277A 31C0           <1>             xor    eax, eax
0001277C F9             <1>             stc
                        <1> loc_swp_retn:
0001277D 89EC           <1>             mov    esp, ebp
0001277F 5D             <1>             pop    ebp
                        <1>
                        <1>             ;mov   esi, FindFile_Drv
00012780 BE[388B0100]   <1>             mov    esi, FindFile_Name ; 12/10/2016
00012785 C3             <1>             retn
                        <1>
                        <1> loc_swp_fchar_next:
00012786 46             <1>             inc    esi
00012787 EBE9           <1>             jmp    short loc_swp_fchar
                        <1>
                        <1> loc_swp_parse_path_name:
00012789 BF[F68A0100]   <1>             mov    edi, FindFile_Drv
0001278E E8AF8EFFFF     <1>             call   parse_path_name
00012793 72E8           <1>             jc     short loc_swp_retn
                        <1>
                        <1> loc_swp_checkfile_name:
00012795 803D[488E0100]00 <1>           cmp    byte [SWP_Mode], 0
0001279C 761E           <1>             jna    short loc_swp_drv
                        <1>
                        <1>             ; 10/10/2016 (valid file name checking)
0001279E BE[388B0100]   <1>             mov    esi, FindFile_Name
000127A3 803E20         <1>             cmp    byte [esi], 20h
000127A6 76D2           <1>             jna    short loc_swp_xor_retn
                        <1>
                        <1>             ; 16/10/2016
000127A8 C605[478E0100]00 <1>           mov    byte [SWP_inv_fname], 0 ; reset
```

```
 977                            <1>             ; esi = file name address (ASCIIZ)
 978 000127AF E87B70FFFF        <1>             call   check_filename
 979 000127B4 7306              <1>             jnc    short loc_swp_drv
 980                            <1>
 981 000127B6 FE05[478E0100]    <1>             inc    byte [SWP_inv_fname] ; set
 982                            <1> loc_swp_drv:
 983 000127BC 8A35[56820100]    <1>             mov    dh, [Current_Drv]
 984                            <1>                ;mov       [RUN_CDRV], dh
 985                            <1>
 986 000127C2 8A15[F68A0100]    <1>             mov    dl, [FindFile_Drv]
 987                            <1>                ;cmp       dl, dh
 988 000127C8 3A15[56820100]    <1>             cmp    dl, [Current_Drv]
 989 000127CE 740D              <1>             je     short loc_swp_change_directory
 990                            <1>
 991 000127D0 FE05[4B8E0100]    <1>             inc    byte [SWP_DRV_chg]
 992 000127D6 E8F358FFFF        <1>             call   change_current_drive
 993 000127DB 72A0              <1>             jc     short loc_swp_retn ; eax = error code
 994                            <1>             ; eax = 0
 995                            <1>
 996                            <1> loc_swp_change_directory:
 997 000127DD 803D[F78A0100]21  <1>             cmp    byte [FindFile_Directory], 21h
 998 000127E4 F5                <1>             cmc
 999 000127E5 7396              <1>             jnc    short loc_swp_retn
1000                            <1>
1001 000127E7 FE05[4B8E0100]    <1>             inc    byte [SWP_DRV_chg]
1002 000127ED FE05[23380100]    <1>             inc    byte [Restore_CDIR]
1003 000127F3 BE[F78A0100]      <1>             mov    esi, FindFile_Directory
1004 000127F8 8A25[498E0100]    <1>             mov    ah, [SWP_Mode+1]
1005 000127FE E82988FFFF        <1>             call   change_current_directory
1006 00012803 0F8274FFFFFF      <1>             jc     loc_swp_retn ; eax = error code
1007                            <1>
1008                            <1> loc_swp_change_prompt_dir_string:
1009                            <1>             ; esi = PATH_Array
1010                            <1>             ; eax = Current Directory First Cluster
1011                            <1>             ; edi = Logical DOS Drive Description Table
1012 00012809 E84387FFFF        <1>             call   change_prompt_dir_str
1013 0001280E 29C0              <1>             sub    eax, eax ; 0
1014 00012810 E968FFFFFF        <1>             jmp    loc_swp_retn
1015                            <1>
1016                            <1> reset_working_path:
1017                            <1>             ; 06/10/2016 - TRDOS 386 (TRDOS v2.0)
1018                            <1>             ;
1019                            <1>             ; TRDOS v1.0 (DIR.ASM, "proc_reset_working_path")
1020                            <1>             ; 05/02/2011 - 08/02/2011
1021                            <1>             ;
1022                            <1>             ; Restores current drive and directory
1023                            <1>             ;
1024                            <1>             ; INPUT: none
1025                            <1>             ; OUTPUT: DL = SWP_DRV, EAX = 0 -> OK
1026                            <1>             ;
1027                            <1>             ;    AX = 0 -> ESI = Logical Dos Drv Desc. Table
1028                            <1>             ;
1029                            <1>             ;    EAX, EBX, ECX, EDX, ESI, EDI will be changed
1030                            <1>             ;
1031                            <1>
1032                            <1>
1033 00012815 31C0              <1>             xor    eax, eax
1034 00012817 48                <1>             dec    eax
1035                            <1>
1036 00012818 668B15[4A8E0100]  <1>             mov    dx, [SWP_DRV]
1037 0001281F 08F6              <1>             or     dh, dh
1038 00012821 742E              <1>             jz     short loc_rwp_return
1039                            <1>
1040 00012823 3A15[56820100]    <1>             cmp    dl, [Current_Drv]
1041 00012829 7407              <1>             je     short loc_rwp_restore_cdir
1042                            <1> loc_rwp_restore_cdrv:
1043 0001282B E89E58FFFF        <1>             call   change_current_drive
1044 00012830 EB10              <1>             jmp    short loc_rwp_restore_ok
1045                            <1> loc_rwp_restore_cdir:
1046 00012832 31DB              <1>             xor    ebx, ebx
1047 00012834 88D7              <1>             mov    bh, dl
1048 00012836 BE00010900        <1>             mov    esi, Logical_DOSDisks
1049 0001283B 01DE              <1>             add    esi, ebx
1050                            <1>
1051 0001283D E84359FFFF        <1>             call   restore_current_directory
1052                            <1>
1053                            <1> loc_rwp_restore_ok:
1054 00012842 668B15[4A8E0100]  <1>             mov    dx, [SWP_DRV]
1055 00012849 31C0              <1>             xor    eax, eax
1056 0001284B 66A3[4B8E0100]    <1>             mov    [SWP_DRV_chg], ax
1057                            <1> loc_rwp_return:
1058 00012851 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 00012852 57                       <1>             push   edi
1088 00012853 56                       <1>             push   esi
1089 00012854 AC                       <1>             lodsb
1090 00012855 3C20                     <1>             cmp    al, 20h
1091 00012857 762A                     <1>             jna    short pass_gfn_ext
1092 00012859 56                       <1>             push   esi
1093 0001285A AA                       <1>             stosb
1094 0001285B B907000000               <1>             mov    ecx, 7
1095                                    <1> loc_gfn_next_char:
1096 00012860 AC                       <1>             lodsb
1097 00012861 3C20                     <1>             cmp    al, 20h
1098 00012863 7603                     <1>             jna    short pass_gfn_fn
1099 00012865 AA                       <1>             stosb
1100 00012866 E2F8                     <1>             loop   loc_gfn_next_char
1101                                    <1> pass_gfn_fn:
1102 00012868 5E                       <1>             pop    esi
1103 00012869 83C607                   <1>             add    esi, 7
1104 0001286C AC                       <1>             lodsb
1105 0001286D 3C20                     <1>             cmp    al, 20h
1106 0001286F 7612                     <1>             jna    short pass_gfn_ext
1107 00012871 B42E                     <1>             mov    ah, '.'
1108 00012873 86E0                     <1>             xchg   ah, al
1109 00012875 66AB                     <1>             stosw
1110 00012877 AC                       <1>             lodsb
1111 00012878 3C20                     <1>             cmp    al, 20h
1112 0001287A 7607                     <1>             jna    short pass_gfn_ext
1113 0001287C AA                       <1>             stosb
1114 0001287D AC                       <1>             lodsb
1115 0001287E 3C20                     <1>             cmp    al, 20h
1116 00012880 7601                     <1>             jna    short pass_gfn_ext
1117 00012882 AA                       <1>             stosb
1118                                    <1> pass_gfn_ext:
1119 00012883 30C0                     <1>             xor    al, al
1120 00012885 AA                       <1>             stosb
1121 00012886 5E                       <1>             pop    esi
1122 00012887 5F                       <1>             pop    edi
1123 00012888 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 00012889 C0E002                   <1>             shl    al, 2 ; IRQ number * 4
1144 0001288C 0FB6D8                   <1>             movzx  ebx, al
1145                                    <1>
1146 0001288F 08E4                     <1>             or     ah, ah
1147 00012891 7508                     <1>             jnz    short shintv_1 ; set (for user call service)
1148                                    <1>
1149                                    <1>             ; 18/03/2017
1150 00012893 81C3[20420100]           <1>             add    ebx, IRQ_list ; reset to default interrupt list
1151 00012899 EB06                     <1>             jmp    short shintv_2
1152                                    <1> shintv_1:
1153 0001289B 81C3[C2280100]           <1>             add    ebx, IRQ_u_list
1154                                    <1> shintv_2:
1155 000128A1 8B13                     <1>             mov    edx, [ebx] ; IRQ handler address
1156                                    <1>
1157                                    <1>             ; 03/03/2017
1158 000128A3 D0E0                     <1>             shl    al, 1 ; IRQ number * 8
1159                                    <1>             ; 18/03/2017
1160 000128A5 0FB6F8                   <1>             movzx  edi, al
1161 000128A8 81C7[A87F0100]           <1>             add    edi, idt + (8*32) ; IRQ 0 offset = idt + 256
1162                                    <1>
1163 000128AE 89D0                     <1>             mov    eax, edx ; IRQ handler address
1164 000128B0 BB00000800               <1>             mov    ebx, 80000h
1165                                    <1>
1166                                    <1>             ;mov   edx, eax
1167 000128B5 66BA008E                 <1>             mov    dx, 8E00h
1168 000128B9 6689C3                   <1>             mov    bx, ax
1169 000128BC 89D8                     <1>             mov    eax, ebx ; /* selector = 0x0008 = cs */
1170                                    <1>                                     ; /* interrupt gate - dpl=0, present */
1171 000128BE AB                       <1>             stosd  ; selector & offset bits 0-15
1172 000128BF 8917                     <1>             mov    [edi], edx ; attributes & offset bits 16-23
1173                                    <1>
1174 000128C1 C3                       <1>             retn
1175                                    <1> IRQ_u_list:
1176                                    <1>             ; 28/02/2017
1177 000128C2 [56090000]               <1>             dd     timer_int
1178 000128C6 [CE100000]               <1>             dd     kb_int
1179 000128CA [380B0000]               <1>             dd     irq2
1180 000128CE [02290100]               <1>             dd     IRQ_service3
1181 000128D2 [0C290100]               <1>             dd     IRQ_service4
1182 000128D6 [16290100]               <1>             dd     IRQ_service5
1183 000128DA [C9510000]               <1>             dd     fdc_int
1184 000128DE [20290100]               <1>             dd     IRQ_service7
1185 000128E2 [C10A0000]               <1>             dd     rtc_int
1186 000128E6 [2A290100]               <1>             dd     IRQ_service9
```

```
1187 000128EA [34290100]          <1>             dd     IRQ_service10
1188 000128EE [3E290100]          <1>             dd     IRQ_service11
1189 000128F2 [48290100]          <1>             dd     IRQ_service12
1190 000128F6 [52290100]          <1>             dd     IRQ_service13
1191 000128FA [7C5B0000]          <1>             dd     hdc1_int
1192 000128FE [A35B0000]          <1>             dd     hdc2_int
1193                              <1>
1194                              <1>             ; 03/03/2017
1195                              <1>             ; 27/02/2017
1196                              <1> IRQ_service3:
1197 00012902 36C605[0E940100]03  <1>             mov    byte [ss:IRQnum], 3
1198 0001290A EB4E                <1>             jmp    short IRQ_service
1199                              <1> IRQ_service4:
1200 0001290C 36C605[0E940100]04  <1>             mov    byte [ss:IRQnum], 4
1201 00012914 EB44                <1>             jmp    short IRQ_service
1202                              <1> IRQ_service5:
1203 00012916 36C605[0E940100]05  <1>             mov    byte [ss:IRQnum], 5
1204 0001291E EB3A                <1>             jmp    short IRQ_service
1205                              <1> IRQ_service7:
1206 00012920 36C605[0E940100]07  <1>             mov    byte [ss:IRQnum], 7
1207 00012928 EB30                <1>             jmp    short IRQ_service
1208                              <1> IRQ_service9:
1209 0001292A 36C605[0E940100]09  <1>             mov    byte [ss:IRQnum], 9
1210 00012932 EB26                <1>             jmp    short IRQ_service
1211                              <1> IRQ_service10:
1212 00012934 36C605[0E940100]0A  <1>             mov    byte [ss:IRQnum], 10
1213 0001293C EB1C                <1>             jmp    short IRQ_service
1214                              <1> IRQ_service11:
1215 0001293E 36C605[0E940100]0B  <1>             mov    byte [ss:IRQnum], 11
1216 00012946 EB12                <1>             jmp    short IRQ_service
1217                              <1> IRQ_service12:
1218 00012948 36C605[0E940100]0C  <1>             mov    byte [ss:IRQnum], 12
1219 00012950 EB08                <1>             jmp    short IRQ_service
1220                              <1> IRQ_service13:
1221 00012952 36C605[0E940100]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 0001295A 1E                  <1>             push   ds
1230 0001295B 06                  <1>             push   es
1231 0001295C 0FA0                <1>             push   fs
1232 0001295E 0FA8                <1>             push   gs
1233                              <1>
1234 00012960 60                  <1>             pushad ; eax,ecx,edx,ebx,esp,ebp,esi,edi
1235 00012961 66B91000            <1>             mov     cx, KDATA
1236 00012965 8ED9                <1>             mov     ds, cx
1237 00012967 8EC1                <1>             mov     es, cx
1238 00012969 8EE1                <1>             mov     fs, cx
1239 0001296B 8EE9                <1>             mov     gs, cx
1240                              <1>
1241 0001296D 0F20D8              <1>             mov    eax, cr3
1242 00012970 A3[0A940100]        <1>             mov    [IRQ_cr3], eax
1243                              <1>
1244 00012975 A1[90810100]        <1>             mov    eax, [k_page_dir]
1245 0001297A 0F22D8              <1>             mov    cr3, eax
1246                              <1>
1247 0001297D A0[0E940100]        <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 00012982 0FB6D8              <1>             movzx  ebx, al
1253 00012985 8A9B[58410100]      <1>             mov    bl, [ebx+IRQenum] ; IRQ (available) index number + 1
1254                              <1>             ; 01/03/2017
1255 0001298B FECB                <1>             dec    bl  ; IRQ index number, 0 to 8
1256 0001298D 0F8807010000        <1>             js     IRQsrv_5 ; not available to use here!?
1257                              <1>             ;
1258 00012993 80BB[D4930100]80    <1>             cmp    byte [ebx+IRQ.method], 80h ; using by a dev or kernel?
1259 0001299A 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 0001299C 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 000129A1 A2[D7030300]        <1>             mov    [u.irqwait], al ; set waiting IRQ flag
1274                              <1>
1275 000129A6 8A83[C2930100]      <1>             mov    al, [ebx+IRQ.owner]
1276 000129AC 20C0                <1>             and    al, al
1277 000129AE 0F84E6000000        <1>             jz     IRQsrv_5 ; it is not owned by a user/proc
1278                              <1>
1279                              <1>             ; 03/03/2017
1280 000129B4 89DA                <1>             mov    edx, ebx
1281 000129B6 C0E202              <1>             shl    dl, 2
1282 000129B9 8B92[E6930100]      <1>             mov    edx, [edx+IRQ.addr] ; S.R.B. or Callback service addr
1283                              <1>
1284 000129BF 8AA3[D4930100]      <1>             mov    ah, [ebx+IRQ.method]
1285 000129C5 F6C401              <1>             test   ah, 1
1286 000129C8 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 000129CA 80E402              <1>             and    ah, 2 ; bit 1, (S.R.B.) counter (auto increment) method
```

```
1292 000129CD 8AA3[DD930100]      <1>             mov    ah, [ebx+IRQ.srb] ; Signal Response Byte value
1293 000129D3 7408                <1>             jz     short IRQsrv_2 ; fixed S.R.B. value
1294                              <1>             ; counter method (auto increment)
1295 000129D5 FEC4                <1>             inc    ah
1296 000129D7 88A3[DD930100]      <1>             mov    [ebx+IRQ.srb], ah ; next (count) number
1297                              <1> IRQsrv_2:
1298 000129DD 8822                <1>             mov    [edx], ah ; put S.R.B. val to the user's S.R.B. addr
1299 000129DF C605[D7030300]00    <1>             mov    byte [u.irqwait], 0 ; clear waiting IRQ flag
1300                              <1>
1301 000129E6 3A05[B3030300]      <1>             cmp    al, [u.uno]
1302 000129EC 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 000129F2 B202                <1>             mov    dl, 2 ; priority, 2 = event (high)
1307 000129F4 E837FAFFFF          <1>             call   set_run_sequence
1308                              <1>
1309                              <1>             ; [u.irqwait] = waiting IRQ number for callback service
1310                              <1>
1311 000129F9 E99C000000          <1>             jmp    IRQsrv_5
1312                              <1> IRQsrv_4:
1313 000129FE 3A05[B3030300]      <1>             cmp    al, [u.uno]  ; is the owner is current user/process?
1314 00012A04 75EC                <1>             jne    short IRQsrv_3 ; no !
1315                              <1>
1316                              <1>             ; Check if an IRQ callback service already in progress
1317 00012A06 803D[D8030300]00    <1>             cmp    byte [u.r_lock], 0
1318 00012A0D 0F8787000000        <1>             ja     IRQsrv_5 ; nothing to do !
1319                              <1>                             ; (we need to complete prev callback)
1320 00012A13 803D[D4030300]00    <1>             cmp    byte [u.t_lock], 0
1321 00012A1A 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 00012A1C C605[D7030300]00    <1>             mov    byte [u.irqwait], 0 ; reset/clear waiting IRQ flag
1326                              <1>
1327 00012A23 FE05[D8030300]      <1>             inc    byte [u.r_lock] ; 'IRQ callback service in progress' flag
1328                              <1>
1329 00012A29 8A0D[5B030300]      <1>             mov    cl, [sysflg]   ; (system call) mode flag (kernel/user)
1330 00012A2F 880D[D9030300]      <1>             mov    [u.r_mode], cl ; system mode (0) or user mode (FFh)
1331                              <1>
1332                              <1>             ;
1333 00012A35 8B2D[2C810100]      <1>             mov    ebp, [tss.esp0] ; kernel stack address (for ring 0)
1334 00012A3B 83ED14              <1>             sub    ebp, 20             ; eip, cs, eflags, esp, ss
1335 00012A3E 892D[5C030300]      <1>             mov    [u.sp], ebp
1336 00012A44 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 00012A4A 8B44241C            <1>             mov    eax, [esp+28] ; pushed eax
1341 00012A4E A3[64030300]        <1>             mov    [u.r0], eax
1342                              <1>
1343 00012A53 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 00012A58 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),
1355                              <1>                             ; ESP (u), SS (UDATA)
1356                              <1>
1357 00012A5F 8B4510              <1>             mov    eax, [ebp+16]; SS (UDATA)
1358 00012A62 89E6                <1>             mov    esi, esp
1359 00012A64 50                  <1>             push   eax
1360 00012A65 50                  <1>             push   eax
1361 00012A66 89E7                <1>             mov    edi, esp
1362 00012A68 893D[60030300]      <1>             mov    [u.usp], edi
1363 00012A6E B908000000          <1>             mov    ecx, ((ESPACE/4) - 4) ; except DS, ES, FS, GS
1364 00012A73 F3A5                <1>             rep    movsd
1365 00012A75 B104                <1>             mov    cl, 4
1366 00012A77 F3AB                <1>             rep    stosd
1367 00012A79 893D[5C030300]      <1>             mov    [u.sp], edi
1368 00012A7F 89EE                <1>             mov    esi, ebp
1369 00012A81 B105                <1>             mov    cl, 5 ; EIP (u), CS (UCODE), EFLAGS (u), ESP (u), SS (UDATA)
1370 00012A83 F3A5                <1>             rep    movsd
1371                              <1>             ;
1372                              <1>
1373 00012A85 8B0D[B8030300]      <1>             mov    ecx, [u.pgdir]
1374 00012A8B 890D[0A940100]      <1>             mov    [IRQ_cr3], ecx
1375                              <1>
1376                              <1> set_IRQ_callback_addr:
1377                              <1>             ;
1378                              <1>             ; This routine sets return address
1379                              <1>             ; to start of user's interrupt
1380                              <1>             ; service (callback) address
1381                              <1>             ;
1382                              <1>             ; INPUT:
1383                              <1>             ;     EDX = callback routine/service address
1384                              <1>             ;           (virtual, not physical address!)
1385                              <1>             ;     [u.sp] = kernel stack, points to
1386                              <1>             ;              user's EIP,CS,EFLAGS,ESP,SS
1387                              <1>             ;              registers.
1388                              <1>             ; OUTPUT:
1389                              <1>             ;     EIP (user) = callback (service) address
1390                              <1>             ;     CS (user) = UCODE
1391                              <1>             ;     EFLAGS (user) = flags before callback
1392                              <1>             ;       ESP (user) = ESP-4 (user, before callback)
1393                              <1>             ;     [ESP](user) = EIP (user) before callback
1394                              <1>             ;
1395                              <1>             ; Note: If CPU was in user mode while entering
1396                              <1>             ;     the timer interrupt service routine,
```

```
1397                                    <1>             ;       'IRET' will get return to callback routine
1398                                    <1>             ;       immediately. If CPU was in system/kernel mode
1399                                    <1>             ;       'iret' will get return to system call and
1400                                    <1>             ;       then, callback routine will be return address
1401                                    <1>             ;       from system call. (User's callback/service code
1402                                    <1>             ;       will be able to return to normal return address
1403                                    <1>             ;       via a 'sysrele' system call at the end.)
1404                                    <1>             ;
1405                                    <1>             ; Note: User's IRQ callback service code must be ended
1406                                    <1>             ;       with a 'sysrele' system call !
1407                                    <1>             ;
1408                                    <1>             ;       For example:
1409                                    <1>             ;
1410                                    <1>             ;       audio_IRQ_callback:
1411                                    <1>             ;           ...
1412                                    <1>             ;           <load DMA buffer with audio data>
1413                                    <1>             ;           ...
1414                                    <1>             ;           mov eax, 39 ; 'sysrele'
1415                                    <1>             ;           int 40h ; TRDOS 386 system call (interrupt)
1416                                    <1>             ;
1417                                    <1>
1418                                    <1>             ;mov   edx, [edx+IRQ.addr] ; Callback service address
1419                                    <1>             ;                          ; (Virtual address)
1420                                    <1>
1421 00012A91 8B2D[5C030300]            <1>             mov    ebp, [u.sp]; kernel's stack, points to EIP (user)
1422 00012A97 895500                   <1>             mov    [ebp], edx
1423                                    <1> IRQsrv_5:
1424                                    <1>             ; EOI & return
1425                                    <1>             ; 01/08/2020
1426                                    <1>             ; 11/06/2017
1427                                    <1>             ; 10/06/2017
1428                                    <1>             ;mov   al, [IRQnum]
1429 00012A9A B020                     <1>             mov    al, 20h ; 01/08/2020
1430 00012A9C FA                       <1>             cli
1431                                    <1>             ;cmp   al, 7
1432 00012A9D 803D[0E940100]07         <1>             cmp    byte [IRQnum], 7 ; 01/08/2020
1433 00012AA4 7602                     <1>             jna    short IRQsrv_6
1434                                    <1>             ;
1435                                    <1>             ;;mov  al, EOI      ; end of interrupt
1436                                    <1>             ;mov   al, 20h ; 01/08/2020
1437                                    <1>             ;cli         ; disable interrupts till stack cleared
1438                                    <1>             ;out   INTB00, al ; For controll2 #2
1439 00012AA6 E6A0                     <1>             out    0A0h, al
1440                                    <1> IRQsrv_6:
1441                                    <1>             ;mov   byte [IRQnum], 0 ; reset
1442                                    <1>             ;;mov  al, EOI      ; end of interrupt
1443                                    <1>             ;mov   al, 20h ; 01/08/2020
1444                                    <1>             ;cli         ; disable interrupts till stack cleared
1445                                    <1>             ;out   INTA00, al ; end of interrupt to 8259 - 1
1446 00012AA8 E620                     <1>             out    20h, al
1447                                    <1> IRQsrv_7:
1448                                    <1>             ;; 13/06/2017
1449                                    <1>             ;or    word [ebp+8], 200h ; force enabling interrupts
1450                                    <1>             ;
1451 00012AAA 8B0D[0A940100]           <1>             mov    ecx, [IRQ_cr3]      ; previous content of cr3 register
1452 00012AB0 0F22D9                   <1>             mov    cr3, ecx     ; restore cr3 register content
1453                                    <1>             ;
1454 00012AB3 61                       <1>             popad ; edi,esi,ebp,(icrement esp by 4),ebx,edx,ecx,eax
1455                                    <1>             ;
1456 00012AB4 0FA9                     <1>             pop    gs
1457 00012AB6 0FA1                     <1>             pop    fs
1458 00012AB8 07                       <1>             pop    es
1459 00012AB9 1F                       <1>             pop    ds
1460                                    <1>             ;
1461 00012ABA CF                       <1>             iretd  ; return from interrupt
1462                                    <1>
1463                                    <1> get_device_number:
1464                                    <1>             ; 08/10/2016
1465                                    <1>             ; 07/10/2016 - TRDOS 386 (TRDOS v2.0)
1466                                    <1>             ;
1467                                    <1>             ; This procedure compares name of requested
1468                                    <1>             ; device with kernel device names and
1469                                    <1>             ; installable device names. If names match,
1470                                    <1>             ; the relevant device index (entry) number
1471                                    <1>             ; will be returned the caller (sysopen)
1472                                    <1>             ; for the requested device.
1473                                    <1>             ;
1474                                    <1>             ; NOTE: Installable device drivers must
1475                                    <1>             ; be loaded before using 'sysopen'
1476                                    <1>             ; (opendev) system call.
1477                                    <1>             ;
1478                                    <1>             ; INPUT:
1479                                    <1>             ;    ESI = device name address (ASCIIZ)
1480                                    <1>             ;         (in kernel's memory space)
1481                                    <1>             ;    max name length = 8 without '/dev/')
1482                                    <1>             ;    Device name will be capitalized
1483                                    <1>             ;    and if there is, '/dev/' will be
1484                                    <1>             ;    removed from name before comparising)
1485                                    <1>             ;
1486                                    <1>             ; OUTPUT:
1487                                    <1>             ;    cf = 0 ->
1488                                    <1>             ;      EAX (AL) = device entry/index number
1489                                    <1>             ;    cf = 1 -> device not found (installed)
1490                                    <1>             ;              or invalid device name
1491                                    <1>             ;              (AL=0)
1492                                    <1>             ;    device_name = device name address (asciiz)
1493                                    <1>             ;      ;
1494                                    <1>             ; Modified registers: EAX, EBX, ESI, EDI
1495                                    <1>
1496 00012ABB BF[4D8E0100]             <1>             mov    edi, device_name
1497 00012AC0 E805010000               <1>             call   lodsb_capitalize
1498 00012AC5 88C4                     <1>             mov    ah, al
1499 00012AC7 3C2F                     <1>             cmp    al, '/'
1500 00012AC9 750E                     <1>             jne    short gdn_1
1501 00012ACB BF[4D8E0100]             <1>             mov    edi, device_name
```

```
1502 00012AD0 E8F5000000          <1>             call   lodsb_capitalize
1503                              <1> gdn_0:
1504 00012AD5 20C0                <1>             and    al, al ; 0 ?
1505 00012AD7 7420                <1>             jz     short gdn_err ; null name after '/'
1506                              <1> gdn_1:
1507 00012AD9 3C44                <1>             cmp    al, 'D'
1508 00012ADB 7517                <1>             jne    short gdn_2
1509 00012ADD E8E8000000          <1>             call   lodsb_capitalize
1510 00012AE2 3C45                <1>             cmp    al, 'E'
1511 00012AE4 750E                <1>             jne    short gdn_2
1512 00012AE6 E8DF000000          <1>             call   lodsb_capitalize
1513 00012AEB 3C56                <1>             cmp    al, 'V'
1514 00012AED 7505                <1>             jne    short gdn_2
1515 00012AEF AC                  <1>             lodsb
1516 00012AF0 3C2F                <1>             cmp    al, '/'
1517 00012AF2 740D                <1>             je     short gdn_4
1518                              <1> gdn_2:
1519 00012AF4 80FC2F              <1>             cmp    ah, '/'
1520 00012AF7 750F                <1>             jne    short gdn_5
1521                              <1> gdn_err:
1522                              <1>             ; invalid device name or device not found
1523 00012AF9 31C0                <1>             xor    eax, eax ; 0
1524 00012AFB F9                  <1>             stc
1525 00012AFC C3                  <1>             retn
1526                              <1> gdn_3:
1527 00012AFD 3C2F                <1>             cmp    al, '/'
1528 00012AFF 7507                <1>             jne    short gdn_5
1529                              <1> gdn_4:
1530 00012B01 BF[4D8E0100]        <1>             mov    edi, device_name
1531 00012B06 EB04                <1>             jmp    short gdn_6
1532                              <1> gdn_5:
1533 00012B08 3C00                <1>             cmp    al, 0
1534 00012B0A 7419                <1>             je     short gdn_7
1535                              <1> gdn_6:
1536 00012B0C E8B9000000          <1>             call lodsb_capitalize
1537 00012B11 81FF[558E0100]      <1>             cmp    edi, device_name + 8
1538 00012B17 72E4                <1>             jb     short gdn_3
1539 00012B19 3C00                <1>             cmp    al, 0
1540 00012B1B 75DC                <1>             jne    short gdn_err
1541 00012B1D 81FF[4E8E0100]      <1>             cmp    edi, device_name + 1
1542 00012B23 76D4                <1>             jna    short gdn_err ; null name after '/'
1543                              <1> gdn_7:
1544 00012B25 AA                  <1>             stosb
1545                              <1>             ; zero padding ("NAME",0,0,0,0)
1546 00012B26 81FF[558E0100]      <1>             cmp    edi, device_name + 8
1547 00012B2C 72F7                <1>             jb     short gdn_7
1548                              <1> gdn_8:
1549                              <1>             ; search for kernel device names
1550 00012B2E BE[4D8E0100]        <1>             mov    esi, device_name
1551 00012B33 BF[3E3F0100]        <1>             mov    edi, KDEV_NAME
1552 00012B38 31C0                <1>             xor    eax, eax
1553                              <1> gdn_9:
1554 00012B3A A7                  <1>             cmpsd
1555 00012B3B 7505                <1>             jne    short gdn_10
1556 00012B3D A7                  <1>             cmpsd
1557 00012B3E 7503                <1>             jne    short gdn_11
1558 00012B40 EB2B                <1>             jmp    short gdn_17 ; match
1559                              <1> gdn_10:
1560 00012B42 A7                  <1>             cmpsd  ; add esi, 4 & add edi, 4
1561                              <1> gdn_11:
1562 00012B43 BE[4D8E0100]        <1>             mov    esi, device_name
1563 00012B48 FEC0                <1>             inc    al
1564 00012B4A 3C16                <1>             cmp    al, NumOfKernelDevNames
1565 00012B4C 72EC                <1>             jb     short gdn_9
1566                              <1> gdn_12:
1567                              <1>             ; search for installable device names
1568                              <1>             ; esi = offset device_name
1569 00012B4E BF[788E0100]        <1>             mov    edi, IDEV_NAME
1570 00012B53 28C0                <1>             sub    al, al ; 0
1571                              <1> gdn_13:
1572 00012B55 A7                  <1>             cmpsd
1573 00012B56 7505                <1>             jne    short gdn_14
1574 00012B58 A7                  <1>             cmpsd
1575 00012B59 7503                <1>             jne    short gdn_15
1576 00012B5B EB3F                <1>             jmp    short gdn_19 ; match
1577                              <1> gdn_14:
1578 00012B5D A7                  <1>             cmpsd  ; add esi, 4 & add edi, 4
1579                              <1> gdn_15:
1580 00012B5E BE[4D8E0100]        <1>             mov    esi, device_name
1581 00012B63 FEC0                <1>             inc    al
1582 00012B65 3C08                <1>             cmp    al, NumOfInstallableDevices
1583 00012B67 72EC                <1>             jb     short gdn_13
1584                              <1>
1585                              <1> gdn_16:     ; error: invalid device name (not found) !
1586 00012B69 30C0                <1>             xor    al, al
1587 00012B6B F9                  <1>             stc
1588 00012B6C 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 00012B6D 89C3                <1>             mov    ebx, eax ; < 256
1597 00012B6F 8A83[EE3F0100]      <1>             mov    al, [KDEV_NUMBER+ebx]
1598                              <1>
1599                              <1>             ; check if empty dev entry in the list
1600 00012B75 80B8[FC8F0100]00    <1>             cmp    byte [DEV_OPENMODE+eax], 0
1601 00012B7C 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 00012B7E 88C3                <1>             mov    bl, al
```

```
1607 00012B80 83EF08              <1>             sub    edi, 8 ; kernel device name address (data)
1608 00012B83 66C1E302            <1>             shl    bx, 2
1609 00012B87 89BB[1A900100]      <1>             mov    [DEV_NAME_PTR+ebx], edi ; (all) device names
1610 00012B8D 8A98[44410100]      <1>             mov    bl, [KDEV_ACCESS+eax] ; kernel dev list (data)
1611 00012B93 8898[488F0100]      <1>             mov    [DEV_ACCESS+eax], bl ; (all) device list (bss)
1612                              <1> gdn_18:
1613 00012B99 FEC0                <1>             inc    al ; 1 to NumOfKernelDevNames (<=7Fh)
1614                              <1>             ; eax = device index/entry number
1615 00012B9B 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 00012B9C 89C3                <1>             mov    ebx, eax
1622 00012B9E 80C316              <1>             add    bl, NumOfKernelDevices    ; < NUMOFDEVICES
1623                              <1>
1624                              <1>             ; check if empty dev entry in the list
1625 00012BA1 80BB[FC8F0100]00    <1>             cmp    byte [DEV_OPENMODE+ebx], 0
1626 00012BA8 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 00012BAA 83EF08              <1>             sub    edi, 8 ; installable device name address
1631 00012BAD 66C1E302            <1>             shl    bx, 2 ;*4
1632 00012BB1 89BB[1A900100]      <1>             mov    [DEV_NAME_PTR+ebx], edi ; (all) device names
1633 00012BB7 66C1EB02            <1>             shr    bx, 2
1634 00012BBB 8A80[C08E0100]      <1>             mov    al, [IDEV_FLAGS+eax] ; installable dev list
1635 00012BC1 8883[488F0100]      <1>             mov    [DEV_ACCESS+ebx], al ; (all) device list
1636                              <1> gdn_20:
1637 00012BC7 88D8                <1>             mov    al, bl
1638                              <1>             ; eax = device index/entry number ; < NUMOFDEVICES
1639 00012BC9 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 00012BCA AC                  <1>       lodsb
1650 00012BCB 3C61                <1>       cmp al, 61h
1651 00012BCD 7206                <1>       jb  short lodsb_cap_retn
1652 00012BCF 3C7A                <1>       cmp al, 7Ah
1653 00012BD1 7702                <1>       ja  short lodsb_cap_retn
1654 00012BD3 24DF                <1>       and al, 0DFh
1655                              <1> lodsb_cap_retn:
1656 00012BD5 AA                  <1>       stosb
1657 00012BD6 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 00012BD7 89C3                <1>       mov    ebx, eax
1676 00012BD9 66C1E302            <1>       shl    bx, 2 ; *4
1677                              <1>
1678 00012BDD F6C580              <1>       test   ch, 80h ; bit 7, installable device driver flag
1679 00012BE0 7406                <1>       jz     short d_open_2 ; Kernel device
1680                              <1>       ; installable device
1681                              <1> d_open_1:
1682 00012BE2 FFA3[C48E0100]      <1>         jmp  dword [ebx+IDEV_OADDR-4]
1683                              <1> d_open_2:
1684 00012BE8 FFA3[00400100]      <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 00012BEE 89C3                <1>       mov    ebx, eax
1703 00012BF0 66C1E302            <1>       shl    bx, 2 ; *4
1704                              <1>
1705 00012BF4 F6C580              <1>       test   ch, 80h ; bit 7, installable device driver flag
1706 00012BF7 7406                <1>       jz     short d_close_2 ; Kernel device
1707                              <1>       ; installable device
1708                              <1> d_close_1:
1709 00012BF9 FFA3[E48E0100]      <1>         jmp  dword [ebx+IDEV_CADDR-4]
1710                              <1> d_close_2:
1711 00012BFF FFA3[50400100]      <1>       jmp    dword [ebx+KDEV_CADDR-4]
```

```
                                <1>
                                <1> rnull:
                                <1>       ; 07/10/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>       ; read null (read from null device)
00012C05 C3                     <1>       retn
                                <1>
                                <1> wnull:
                                <1>       ; 07/10/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>       ; write null (write to null device)
00012C06 C3                     <1>       retn
                                <1>
                                <1> dev_IRQ_service:
                                <1>       ; 12/05/2017
                                <1>       ; 13/04/2017
                                <1>       ; 27/02/2017 - TRDOS 386 (TRDOS v2.0)
                                <1>       ; INPUT ->
                                <1>       ;     AL = IRQ Number (0 to 15)
                                <1>       ;
00012C07 53                     <1>       push   ebx
00012C08 0FB6D8                 <1>       movzx  ebx, al
00012C0B C0E302                 <1>       shl    bl, 2 ; * 4
00012C0E 8B9B[82930100]         <1>       mov    ebx, [ebx+DEV_INT_HNDLR]
00012C14 21DB                   <1>       and    ebx, ebx
00012C16 7404                   <1>         jz   short dIRQ_s_retn
00012C18 50                     <1>       push   eax
                                <1>
00012C19 FFD3                   <1>       call   ebx
                                <1>
00012C1B 58                     <1>       pop    eax
                                <1> dIRQ_s_retn:
00012C1C 5B                     <1>       pop    ebx
00012C1D C3                     <1>       retn
                                <1>
                                <1>
                                <1> set_dev_IRQ_service:
                                <1>       ; 13/04/2017 - TRDOS 386 (TRDOS v2.0)
                                <1>       ;
                                <1>       ; Set Device Interrupt Service
                                <1>       ;
                                <1>       ; INPUT ->
                                <1>       ;     AL = IRQ Number
                                <1>       ;     EBX = Hardware Interrupt Service Address
                                <1>       ;
                                <1>       ; Note: There is not a validation check here
                                <1>       ;     because this procedure is called by
                                <1>       ;     TRDOS 386 kernel !
                                <1>       ;       (Even if a device driver does not exist
                                <1>       ;     this setting may be used by sysaudio
                                <1>       ;     and other system calls for hardware
                                <1>       ;     components which use IRQ method for I/O.)
                                <1>       ;
                                <1>       ;push  esi
00012C1E 0FB6F0                 <1>       movzx  esi, al
00012C21 66C1E602               <1>       shl    si, 2 ; * 4
00012C25 899E[82930100]         <1>       mov    [esi+DEV_INT_HNDLR], ebx
                                <1>       ;pop   esi
00012C2B C3                     <1>       retn
                                <1>
                                <1>
                                <1> sysaudio: ; AUDIO FUNCTIONS
                                <1>       ; 28/07/2020
                                <1>       ; 27/07/2020
                                <1>       ; 10/10/2017
                                <1>       ; 22/06/2017
                                <1>       ; 28/05/2017, 04/06/2017, 05/06/2017, 10/06/2017
                                <1>       ; 01/05/2017, 12/05/2017, 15/05/2017, 20/05/2017
                                <1>       ; 21/04/2017, 22/04/2017, 23/04/2017, 24/04/2017
                                <1>       ; 10/04/2017, 13/04/2017, 14/04/2017, 16/04/2017
                                <1>       ; 03/04/2017 (VIA VT8237R)
                                <1>       ; 01/04/2016 (trdosk6.s -> tdosk8.s)
                                <1>       ; 16/05/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>       ;
                                <1>       ; Inputs:
                                <1>       ;
                                <1>       ;     BH = 0 -> Beep (PC Speaker)
                                <1>       ;          BL = Duration Counter (1 for 1/64 second)
                                <1>       ;          CX = Frequency Divisor (1193180/Frequency)
                                <1>       ;             (1331 for 886 Hz)
                                <1>       ;
                                <1>       ;     01/04/2017
                                <1>       ;
                                <1>       ;     BH = 1 -> DETECT (& ENABLE) AUDIO DEVICE
                                <1>       ;          BL = 0 : PC SPEAKER
                                <1>       ;               1 : SOUND BLASTER 16
                                <1>       ;               2 : INTEL AC'97
                                <1>       ;               3 : VIA VT8237R (VT8233)
                                <1>       ;               4 : INTEL HDA
                                <1>       ;            5-FEh : unknown/invalid
                                <1>       ;              ; 04/06/2017
                                <1>       ;             FFh : Get current audio device id
                                <1>       ;
                                <1>       ;     BH = 2 -> ALLOCATE AUDIO BUFFER (for user)
                                <1>       ;           ECX = Audio Buffer Size (must be equal to
                                <1>       ;                 the half of DMA buffer size)
                                <1>       ;           EDX = Virtual Address of the buffer
                                <1>       ;                 (This is not DMA buffer!)
                                <1>       ;
                                <1>       ;     BH = 3 -> INITIALIZE AUDIO DEVICE
                                <1>       ;          BL = 0,2 -> for Signal Response Byte
                                <1>       ;           CL = Signal Response Byte Value (fixed)
                                <1>       ;                        if BL = 0
                                <1>       ;                auto increment of S.R.B. value
                                <1>       ;                        if BL = 2
                                <1>       ;             EDX = Signal Response (Return) Byte Address
                                <1>       ;
```

```
1817                    <1>     ;           BL = 1 for CallBack Method
1818                    <1>     ;             EDX = CallBack Service Address (Virtual)
1819                    <1>     ;
1820                    <1>     ;           BL > 2 -> invalid function
1821                    <1>     ;
1822                    <1>     ;          (Audio buffer must be allocated before
1823                    <1>     ;           initialization.)
1824                    <1>     ;
1825                    <1>     ;      BH = 4 -> START TO PLAY
1826                    <1>     ;           BL = Mode
1827                    <1>     ;               Bit 0 = mono/stereo (1 = stereo)
1828                    <1>     ;               Bit 1 = 8 bit / 16 bit (1 = 16 bit)
1829                    <1>     ;           CX = Sampling Rate (Hz)
1830                    <1>     ;
1831                    <1>     ;      BH = 5 -> PAUSE
1832                    <1>     ;           BL = Any
1833                    <1>     ;
1834                    <1>     ;      BH = 6 -> CONTINUE TO PLAY
1835                    <1>     ;           BL = Any
1836                    <1>     ;
1837                    <1>     ;      BH = 7 -> STOP
1838                    <1>     ;           BL = Any
1839                    <1>     ;
1840                    <1>     ;      BH = 8 -> RESET
1841                    <1>     ;           BL = Any
1842                    <1>     ;
1843                    <1>     ;      BH = 9 -> CANCEL (CALLBACK or S.R.B. SERVICE)
1844                    <1>     ;           BL = Any
1845                    <1>     ;
1846                    <1>     ;      BH = 10 -> DEALLOCATE AUDIO BUFFER (for user)
1847                    <1>     ;           BL = Any
1848                    <1>     ;
1849                    <1>     ;      BH = 11 -> SET VOLUME LEVEL
1850                    <1>     ;           BL: (Bit 0 to 6)
1851                    <1>     ;               0 = Master (Playback, Lineout) volume
1852                    <1>     ;           CL = Left Channel Volume
1853                    <1>     ;           CH = Right Channel Volume
1854                    <1>     ;
1855                    <1>     ;           Note: If BL >= 80h (Bit 7 of BL is set),
1856                    <1>     ;           volume level will be set for next playing
1857                    <1>     ;           (actual volume level will not be changed
1858                    <1>     ;           immediately)
1859                    <1>     ;
1860                    <1>     ;      BH = 12 -> DISABLE AUDIO DEVICE
1861                    <1>     ;           (reset audio device and unlink dma buffer)
1862                    <1>     ;           BL = Any
1863                    <1>     ;
1864                    <1>     ;      12/05/2017
1865                    <1>     ;      BH = 13 -> MAP DMA BUFFER TO USER
1866                    <1>     ;          (for direct access to system's dma buffer)
1867                    <1>     ;
1868                    <1>     ;           ECX = map size in bytes
1869                    <1>     ;               (will be rounded up to page borders)
1870                    <1>     ;           EDX = Virtual Address of the buffer
1871                    <1>     ;               (Will be rounded up to page borders)
1872                    <1>     ;
1873                    <1>     ;      05/06/2017
1874                    <1>     ;      04/06/2017
1875                    <1>     ;      BH = 14 -> GET AUDIO DEVICE INFO
1876                    <1>     ;           BL: 0 = Audio Controller Info
1877                    <1>     ;             > 0 = Invalid for now!
1878                    <1>     ;
1879                    <1>     ;      22/06/2017
1880                    <1>     ;      BH = 15 -> GET CURRENT SOUND DATA (for graphics)
1881                    <1>     ;           BL: 0 -> PCM OUT data
1882                    <1>     ;             > 0 -> Invalid for now!
1883                    <1>     ;           ECX = 0 -> Get DMA Buffer Pointer
1884                    <1>     ;              EDX = Not Used
1885                    <1>     ;           ECX > 0 -> Byte count for buffer (EDX)
1886                    <1>     ;               EDX = Buffer Address (Virtual)
1887                    <1>     ;
1888                    <1>     ;      10/10/2017
1889                    <1>     ;      BH = 16 -> UPDATE DMA BUFFER DATA
1890                    <1>     ;                 (by using the Audio Buffer content)
1891                    <1>     ;            BL = 0 : Update dma half buffer in sequence
1892                    <1>     ;                     (automatic destination)
1893                    <1>     ;                 1 : Update 1st half of the dma buffer
1894                    <1>     ;                 2 : Update 2nd half of the dma buffer
1895                    <1>     ;                 3-FEh: Invalid!
1896                    <1>     ;                 FFh = Get current flag value
1897                    <1>     ;                      (Half buffer number -1)
1898                    <1>     ;
1899                    <1>     ;
1900                    <1>     ; Outputs:
1901                    <1>     ;
1902                    <1>     ;      For BH = 0 -> Beep
1903                    <1>     ;          None
1904                    <1>     ;
1905                    <1>     ;      01/04/2017
1906                    <1>     ;
1907                    <1>     ;      For BH = 1 -> DETECT (& ENABLE) AUDIO DEVICE
1908                    <1>     ;          AH = 0 : PC SPEAKER
1909                    <1>     ;               1 : SOUND BLASTER 16
1910                    <1>     ;               2 : INTEL AC'97
1911                    <1>     ;               3 : VIA VT8237R (VT8233)
1912                    <1>     ;               4 : INTEL HDA
1913                    <1>     ;             5-FFh : unknown/invalid
1914                    <1>     ;          AL = mode status
1915                    <1>     ;              bit 0 = mono /stereo (1 = stereo)
1916                    <1>     ;              bit 1 = 8 bit / 16 bit ( 1 = 16 bit)
1917                    <1>     ;          04/06/2017
1918                    <1>     ;          EBX = PCI DEVICE/VENDOR ID (if >0)
1919                    <1>     ;              (BX = VENDOR ID)
1920                    <1>     ;          (if CF = 1 -> Error code in EAX)
1921                    <1>     ;
```

```
1922                                    <1>      ;      For BH = 2 -> ALLOCATE AUDIO BUFFER (for user)
1923                                    <1>      ;          EAX = Physical Address of the buffer
1924                                    <1>      ;          (if CF = 1 -> Error code in EAX)
1925                                    <1>      ;
1926                                    <1>      ;      For BH = 3 -> INITIALIZE AUDIO DEVICE
1927                                    <1>      ;          (if CF = 1 -> Error code in EAX)
1928                                    <1>      ;
1929                                    <1>      ;      For BH = 4 -> START TO PLAY
1930                                    <1>      ;          none (if CF = 1 -> Error code in EAX)
1931                                    <1>      ;
1932                                    <1>      ;      For BH = 5 -> PAUSE
1933                                    <1>      ;          none (if CF = 1 -> Error code in EAX)
1934                                    <1>      ;
1935                                    <1>      ;      For BH = 6 -> CONTINUE TO PLAY
1936                                    <1>      ;          none (if CF = 1 -> Error code in EAX)
1937                                    <1>      ;
1938                                    <1>      ;      For BH = 7 -> STOP
1939                                    <1>      ;          none (if CF = 1 -> Error code in EAX)
1940                                    <1>      ;
1941                                    <1>      ;      For BH = 8 -> RESET
1942                                    <1>      ;          none (if CF = 1 -> Error code in EAX)
1943                                    <1>      ;
1944                                    <1>      ;      For BH = 9 -> CANCEL (CALLBACK or S.R.B. SERVICE)
1945                                    <1>      ;          none (if CF = 1 -> Error code in EAX)
1946                                    <1>      ;
1947                                    <1>      ;      For BH = 10 -> DEALLOCATE AUDIO BUFFER (for user)
1948                                    <1>      ;          none (if CF = 1 -> Error code in EAX)
1949                                    <1>      ;
1950                                    <1>      ;      For BH = 11 -> SET VOLUME LEVEL
1951                                    <1>      ;          none (if CF = 1 -> Error code in EAX)
1952                                    <1>      ;
1953                                    <1>      ;      For BH = 12 -> DISABLE AUDIO DEVICE
1954                                    <1>      ;          none (if CF = 1 -> Error code in EAX)
1955                                    <1>      ;
1956                                    <1>      ;      12/05/2017
1957                                    <1>      ;      For BH = 13 -> MAP DMA BUFFER TO USER
1958                                    <1>      ;          EAX = Physical Address of the buffer
1959                                    <1>      ;          (if CF = 1 -> Error code in EAX)
1960                                    <1>      ;
1961                                    <1>      ;      04/06/2017
1962                                    <1>      ;      For BH = 14 -> GET AUDIO DEVICE INFO
1963                                    <1>      ;      (for BL = 0) ; 05/06/2017
1964                                    <1>      ;          EAX = IRQ Number in AL
1965                                    <1>      ;               Audio Device Number in AH
1966                                    <1>      ;          EBX = DEV/VENDOR ID
1967                                    <1>      ;              (DDDDDDDDDDDDDDDDVVVVVVVVVVVVVVVV)
1968                                    <1>      ;          ECX = BUS/DEV/FN
1969                                    <1>      ;              (00000000BBBBBBBBDDDDDFFF00000000)
1970                                    <1>      ;          EDX = NABMBAR/NAMBAR (for AC97)
1971                                    <1>      ;               (Low word, DX = NAMBAR address)
1972                                    <1>      ;          EDX = Base IO Addr (DX) for SB16 & VT8233
1973                                    <1>      ;          (if CF = 1 -> Error code in EAX)
1974                                    <1>      ;                        (ERR_DEV_NOT_RDY = 15)
1975                                    <1>      ;
1976                                    <1>      ;      22/06/2017
1977                                    <1>      ;      For BH = 15 -> GET CURRENT SOUND DATA
1978                                    <1>      ;                     (for graphics)
1979                                    <1>      ;      (for BL = 0)
1980                                    <1>      ;       If ECX input is 0
1981                                    <1>      ;          EAX = DMA Buffer Current Position (Offset)
1982                                    <1>      ;       If ECX input >  0
1983                                    <1>      ;          EAX = Actual transfer count
1984                                    <1>      ;          (Sound samples will be copied from
1985                                    <1>      ;           Current DMA Buffer Position to EDX
1986                                    <1>      ;           virtual address as EAX bytes.)
1987                                    <1>      ;        ((If CF = 1 ->  Error code in EAX))
1988                                    <1>      ;
1989                                    <1>      ;
1990                                    <1>      ;      10/10/2017
1991                                    <1>      ;      For BH = 16 -> UPDATE DMA BUFFER DATA
1992                                    <1>      ;          EAX = 0, if the updated (or current)
1993                                    <1>      ;                   half buffer is DMA half buffer 1
1994                                    <1>      ;          EAX = 1, if the updated (or current)
1995                                    <1>      ;                   half buffer is DMA half buffer 2
1996                                    <1>      ;          (If CF = 1 -> Error code in EAX)
1997                                    <1>      ;
1998                                    <1>
1999 00012C2C 80FF11                   <1>      cmp    bh, AUDIO1L/4
2000 00012C2F 0F83B7ACFFFF             <1>      jnb    sysret
2001                                    <1>
2002 00012C35 C0E702                   <1>      shl    bh, 2 ; *4
2003 00012C38 0FB6F7                   <1>      movzx  esi, bh
2004                                    <1>
2005                                    <1>      ; 22/04/2017
2006 00012C3B 31C0                     <1>      xor    eax, eax
2007 00012C3D A3[64030300]             <1>      mov    [u.r0], eax  ; 0
2008                                    <1>
2009 00012C42 FF96[4D2C0100]           <1>      call   dword [esi+AUDIO1]
2010                                    <1>      ;jc    error
2011 00012C48 E99FACFFFF               <1>      jmp    sysret
2012                                    <1>
2013 00012C4D [F0230000]                <1> AUDIO1:     dd    beep ; FUNCTION = 0 (bl = Duration Counter
2014                                    <1>                     ;                   cx = Frequency Divisor
2015 00012C51 [912C0100]                <1>      dd     soundc_detect
2016 00012C55 [2D2D0100]                <1>      dd     sound_alloc
2017 00012C59 [EB2D0100]                <1>      dd     soundc_init
2018 00012C5D [A32F0100]                <1>      dd     sound_play
2019 00012C61 [3F300100]                <1>      dd     sound_pause
2020 00012C65 [69300100]                <1>      dd     sound_continue
2021 00012C69 [93300100]                <1>      dd     sound_stop
2022 00012C6D [BC300100]                <1>      dd     soundc_reset
2023 00012C71 [ED300100]                <1>      dd     soundc_cancel
2024 00012C75 [13310100]                <1>      dd     sound_dalloc
2025 00012C79 [3E310100]                <1>      dd     sound_volume
2026 00012C7D [90310100]                <1>      dd     soundc_disable
```

```
2027 00012C81 [02320100]          <1>       dd     sound_dma_map
2028 00012C85 [71320100]          <1>       dd     soundc_info
2029 00012C89 [D0320100]          <1>       dd     sound_data
2030 00012C8D [7D330100]          <1>       dd     sound_update
2031                              <1>
2032                              <1> AUDIO1L      EQU    $ - AUDIO1
2033                              <1>
2034                              <1> soundc_detect:
2035                              <1>       ; FUNCTION = 1
2036                              <1>       ; bl = Audio device type number
2037                              <1>       ; (0= pc speaker, 1 = sound blaster 16, 2 = intel ac97
2038                              <1>       ;  3= via vt823x, 4 = intel HDA, 0FFh= any)
2039                              <1>
2040                              <1>       ; 04/06/2017
2041 00012C91 8A25[11940100]      <1>       mov    ah, [audio_device]
2042 00012C97 80FBFF              <1>       cmp    bl, 0FFh ; get current audio device id
2043 00012C9A 7408                <1>       je     short sysaudio0
2044                              <1>
2045 00012C9C 20E4                <1>       and    ah, ah
2046 00012C9E 741E                <1>       jz     short soundc_get_dev
2047                              <1>
2048 00012CA0 38DC                <1>       cmp    ah, bl
2049 00012CA2 7567                <1>       jne    short soundc_dev_err
2050                              <1>
2051                              <1> sysaudio0:
2052 00012CA4 A0[12940100]        <1>       mov    al, [audio_mode]
2053                              <1> sysaudio1:
2054 00012CA9 A3[64030300]        <1>       mov    [u.r0], eax
2055 00012CAE 8B1D[1C940100]      <1>       mov    ebx, [audio_vendor] ; (DEVICE/VENDOR ID)
2056 00012CB4 8B2D[60030300]      <1>       mov    ebp, [u.usp]
2057 00012CBA 895D10              <1>       mov    [ebp+16], ebx  ; ebx
2058 00012CBD C3                  <1>       retn
2059                              <1>
2060                              <1> soundc_get_dev:
2061                              <1>       ; 28/05/2017
2062                              <1>       ; 03/04/2017, 24/04/2017
2063 00012CBE C605[10940100]00    <1>       mov    byte [audio_pci], 0
2064 00012CC5 80FB03              <1>       cmp    bl, 3 ; VIA VT8233 (VT8237R) Audio Controller & AC97 Codec
2065                              <1>       ;jne   short soundc_get_dev_sb
2066                              <1>       ; 28/05/2017
2067 00012CC8 7220                <1>       jb     short soundc_get_dev_sb
2068 00012CCA 773F                <1>       ja     short soundc_dev_err  ; temporary (28/05/2017)
2069                              <1>       ;
2070 00012CCC E86C180000          <1>       call   DetectVT8233
2071 00012CD1 7238                <1>       jc     short soundc_dev_err
2072                              <1>       ; eax = 0
2073                              <1>
2074                              <1>       ;mov   ebx, [audio_vendor]
2075                              <1>       ; ebx = DEVICE/VENDOR ID
2076                              <1>       ;      DDDDDDDDDDDDDDDDVVVVVVVVVVVVVVVV
2077                              <1>
2078 00012CD3 B003                <1>       mov    al, 3  ; VIA VT8237R (VT3233) Audio Controller
2079 00012CD5 88C4                <1>       mov    ah, al
2080                              <1>
2081                              <1> soundc_get_pci_dev_ok: ; 28/05/2017
2082 00012CD7 FE05[10940100]      <1>       inc    byte [audio_pci] ; = 1
2083                              <1> soundc_get_dev_ok:
2084                              <1>
2085                              <1> soundc_get_dev_sb16_ok:
2086 00012CDD A2[11940100]        <1>       mov    [audio_device], al
2087 00012CE2 8825[12940100]      <1>       mov    [audio_mode], ah ; stereo (bit0), 16 bit (bit1) capability
2088 00012CE8 EBBF                <1>       jmp    short sysaudio1
2089                              <1>
2090                              <1> soundc_get_dev_sb:
2091                              <1>       ; 24/04/2017
2092 00012CEA 80FB01              <1>       cmp    bl, 1 ; Sound Blaster 16
2093 00012CED 750E                <1>       jne    short soundc_get_dev_ich ; 28/05/2017
2094                              <1>       ;
2095 00012CEF E86D1D0000          <1>       call   DetectSB
2096 00012CF4 7215                <1>       jc     short soundc_dev_err
2097 00012CF6 B801030000          <1>       mov    eax, 0301h ; Sound Blaster 16
2098 00012CFB EBE0                <1>       jmp    short soundc_get_dev_sb16_ok
2099                              <1>
2100                              <1> soundc_get_dev_ich:
2101                              <1>       ; 28/05/2017
2102                              <1>       ;cmp   bl, 2 ; Intel AC'97 Audio Controller (ICH)
2103                              <1>       ;jne   short soundc_dev_err ; Temporary  (28/05/2017)
2104                              <1>       ;                           ; (Here will be modified just after
2105                              <1>       ;                           ; new sound card code will be ready!)
2106 00012CFD E82E180000          <1>       call   DetectICH
2107 00012D02 7207                <1>       jc     short soundc_dev_err
2108                              <1>       ;
2109 00012D04 B802030000          <1>       mov    eax, 0302h ; AC'97 (ICH)
2110 00012D09 EBCC                <1>       jmp    short soundc_get_pci_dev_ok
2111                              <1>
2112                              <1> soundc_dev_err:
2113 00012D0B B80F000000          <1>       mov    eax, ERR_DEV_NOT_RDY ; Device not ready !
2114 00012D10 EB0C                <1>       jmp    short sysaudio_err
2115                              <1>
2116                              <1> sound_buff_error:
2117 00012D12 B82E000000          <1>       mov    eax, ERR_BUFFER ; Buffer error !
2118 00012D17 EB05                <1>       jmp    short sysaudio_err
2119                              <1>
2120                              <1> soundc_respond_err:
2121                              <1>       ; ERR_TIME_OUT ; 'time out !' error
2122 00012D19 B819000000          <1>       mov    eax, ERR_DEV_NOT_RESP ; 'device not responding !' error
2123                              <1> sysaudio_err:
2124 00012D1E A3[64030300]        <1>       mov    [u.r0], eax
2125 00012D23 A3[C8030300]        <1>       mov    [u.error], eax
2126 00012D28 E99FABFFFF          <1>       jmp    error
2127                              <1>
2128                              <1> sound_alloc:
2129                              <1>       ; FUNCTION =  2
2130                              <1>       ; ecx = audio buffer size (in bytes)
2131                              <1>       ; edx = audio buffer address (virtual)
```

```
2132                                    <1>       ; 27/07/2020
2133                                    <1>       ; 28/05/2017
2134                                    <1>       ; 01/05/2017, 15/05/2017
2135                                    <1>       ; 21/04/2017, 24/04/2017
2136 00012D2D 803D[10940100]00          <1>       cmp   byte [audio_pci], 0
2137 00012D34 7708                      <1>       ja    short snd_alloc_0
2138                                    <1>       ; Max. 64KB DMA buffer !!!
2139 00012D36 81F900800000              <1>       cmp   ecx, 32768
2140 00012D3C 77D4                      <1>       ja    short sound_buff_error
2141                                    <1> snd_alloc_0:
2142                                    <1>       ; 15/05/2017
2143 00012D3E 81F900100000              <1>       cmp   ecx, 4096 ; PAGE_SIZE
2144 00012D44 72CC                      <1>       jb    short sound_buff_error
2145                                    <1>       ;
2146 00012D46 A1[24940100]              <1>       mov   eax, [audio_buffer] ; audio buffer address (current)
2147 00012D4B 09C0                      <1>       or    eax, eax
2148 00012D4D 7445                      <1>       jz    short snd_alloc_2
2149                                    <1>       ; audio buffer exists !
2150 00012D4F 8A1D[B3030300]            <1>       mov   bl, [u.uno]
2151 00012D55 3A1D[39940100]            <1>       cmp   bl, [audio_user]
2152 00012D5B 0F85FC000000              <1>       jne   sndc_owner_error ; not owner !
2153 00012D61 39D0                      <1>       cmp   eax, edx ; same virtual buffer address ?
2154 00012D63 7508                      <1>       jne   short snd_alloc_1
2155 00012D65 3B0D[2C940100]            <1>       cmp   ecx, [audio_buff_size]
2156 00012D6B 746C                      <1>       je    short snd_alloc_3 ; Nothing to do !
2157                                    <1>                             ; Buffer has been set already!
2158                                    <1> snd_alloc_1:
2159 00012D6D 51                        <1>       push  ecx
2160 00012D6E 52                        <1>       push  edx
2161 00012D6F 89C3                      <1>       mov   ebx, eax ; audio buffer address (current)
2162 00012D71 8B0D[2C940100]            <1>       mov   ecx, [audio_buff_size]
2163 00012D77 E8523AFFFF                <1>       call  deallocate_user_pages
2164 00012D7C 5A                        <1>       pop   edx
2165 00012D7D 59                        <1>       pop   ecx
2166 00012D7E 31C0                      <1>       xor   eax, eax ; 0
2167 00012D80 A3[24940100]              <1>       mov   [audio_buffer], eax  ; 0
2168 00012D85 A3[28940100]              <1>       mov   [audio_p_buffer], eax  ; 0
2169 00012D8A A3[2C940100]              <1>       mov   [audio_buff_size], eax
2170 00012D8F A2[39940100]              <1>       mov   [audio_user], al ; 0
2171                                    <1> snd_alloc_2:
2172 00012D94 89D3                      <1>       mov   ebx, edx
2173                                    <1>       ; 01/05/2017
2174 00012D96 BA00F0FFFF                <1>       mov   edx, ~PAGE_OFF ; truncating page offsets
2175                                    <1>                            ; for aligning to page borders
2176                                    <1>       ;and  eax, edx
2177 00012D9B 21D3                      <1>       and   ebx, edx
2178 00012D9D 21D1                      <1>       and   ecx, edx
2179                                    <1>       ; 15/05/2017
2180                                    <1>       ; EAX = Beginning address (physical)
2181                                    <1>       ; EAX = 0 -> Allocate mem block from the 1st proper aperture
2182                                    <1>       ; ECX = Number of bytes to be allocated
2183 00012D9F E8CF36FFFF                <1>       call  allocate_memory_block
2184 00012DA4 0F8268FFFFFF              <1>       jc    sound_buff_error
2185                                    <1>       ; EAX = Physical address of the allocated memory block
2186                                    <1>       ; ECX = Allocated bytes (as truncated to page border)
2187                                    <1>       ; EBX = Virtual address (as truncated to page border)
2188 00012DAA 50                        <1>       push  eax
2189 00012DAB 53                        <1>       push  ebx
2190 00012DAC 51                        <1>       push  ecx
2191 00012DAD E8113BFFFF                <1>       call  allocate_user_pages
2192 00012DB2 59                        <1>       pop   ecx
2193 00012DB3 5B                        <1>       pop   ebx
2194 00012DB4 58                        <1>       pop   eax
2195 00012DB5 722A                      <1>       jc    short snd_alloc_4  ; insufficient memory, buff error
2196                                    <1>       ; eax = physical address of the user's audio buffer
2197                                    <1>       ; ebx = virtual address of the user's audio buffer
2198                                    <1>       ; ecx = buffer size (in bytes)
2199 00012DB7 A3[28940100]              <1>       mov   [audio_p_buffer], eax
2200 00012DBC 891D[24940100]            <1>       mov   [audio_buffer], ebx
2201 00012DC2 890D[2C940100]            <1>       mov   [audio_buff_size], ecx
2202 00012DC8 8A15[B3030300]            <1>       mov   dl, [u.uno]
2203 00012DCE 8815[39940100]            <1>       mov   [audio_user], dl
2204 00012DD4 A3[64030300]              <1>       mov   [u.r0], eax
2205                                    <1> snd_alloc_3:
2206                                    <1>       ; 27/07/2020
2207 00012DD9 C605[38940100]00          <1>       mov   byte [audio_flag], 0 ; clear dma half buffer flag
2208                                    <1>       ;
2209 00012DE0 C3                        <1>       retn
2210                                    <1> snd_alloc_4:
2211                                    <1>       ; 15/05/2017
2212                                    <1>       ; EAX = Beginning address (physical)
2213                                    <1>       ; ECX = Number of bytes to be deallocated
2214 00012DE1 E89A38FFFF                <1>       call  deallocate_memory_block
2215 00012DE6 E927FFFFFF                <1>       jmp   sound_buff_error  ; insufficient memory, buff error
2216                                    <1>
2217                                    <1> soundc_init:
2218                                    <1>       ; FUNCTION = 3
2219                                    <1>       ; bl = method (0= s.r.b., 1= callback, 2= auto incr s.r.b.)
2220                                    <1>       ; cl = signal response byte (initial or fixed) value
2221                                    <1>       ; edx = signal response byte or callback address
2222                                    <1>       ; 27/07/2020
2223                                    <1>       ; 28/05/2017
2224                                    <1>       ; 12/05/2017, 20/05/2017
2225                                    <1>       ; 22/04/2017, 23/04/2017, 24/04/2017
2226                                    <1>       ; 13/04/2017, 14/04/2017, 16/04/2017, 21/04/2017
2227                                    <1>       ; 03/04/2017, 10/04/2017
2228                                    <1>
2229 00012DEB A0[11940100]              <1>       mov   al, [audio_device]
2230 00012DF0 20C0                      <1>       and   al, al
2231 00012DF2 7549                      <1>       jnz   short sndc_init_6
2232                                    <1>       ;
2233 00012DF4 C605[10940100]00          <1>       mov   byte [audio_pci], 0
2234 00012DFB 52                        <1>       push  edx
2235 00012DFC 53                        <1>       push  ebx
2236 00012DFD 51                        <1>       push  ecx
```

```
2237 00012DFE E85E1C0000          <1>       call   DetectSB
2238 00012E03 7213                <1>       jc     short sndc_init_8
2239 00012E05 66B80103            <1>       mov    ax, 0301h ; Sound Blaster 16
2240 00012E09 EB1E                <1>       jmp    short sndc_init_7
2241                              <1>
2242                              <1> sndc_init_11:
2243                              <1>       ; 28/05/2017
2244 00012E0B E820170000          <1>       call   DetectICH ; Detect AC'97 (ICH) Audio Controller
2245 00012E10 7217                <1>       jc     short sndc_init_7
2246 00012E12 66B80203            <1>       mov    ax, 0302h ; Intel AC'97 Audio Device
2247 00012E16 EB0B                <1>       jmp    short sndc_init_12 ; (PCI device)
2248                              <1>
2249                              <1> sndc_init_8:
2250 00012E18 E820170000          <1>       call   DetectVT8233
2251                              <1>       ;jc    short sndc_init_7
2252 00012E1D 72EC                <1>       jc     sndc_init_11 ; 28/05/2017
2253                              <1>       ; eax = 0
2254 00012E1F B003                <1>       mov    al, 3  ; VIA VT8237R (VT3233) Audio Controller
2255 00012E21 88C4                <1>       mov    ah, al
2256                              <1>
2257                              <1> sndc_init_12:
2258 00012E23 FE05[10940100]      <1>       inc    byte [audio_pci] ; = 1
2259                              <1> sndc_init_7:
2260 00012E29 59                  <1>       pop    ecx
2261 00012E2A 5B                  <1>       pop    ebx
2262 00012E2B 5A                  <1>       pop    edx
2263 00012E2C 0F82D9FEFFFF        <1>       jc     soundc_dev_err
2264                              <1>       ;
2265 00012E32 A2[11940100]        <1>       mov    [audio_device], al
2266 00012E37 8825[12940100]      <1>       mov    [audio_mode], ah ; stereo (bit0), 16 bit (bit1) capability
2267                              <1>
2268                              <1> sndc_init_6:
2269 00012E3D 833D[24940100]00    <1>       cmp    dword [audio_buffer], 0
2270 00012E44 0F86C8FEFFFF        <1>       jna    sound_buff_error
2271                              <1>
2272 00012E4A A0[B3030300]        <1>       mov    al, [u.uno]
2273 00012E4F 8A25[39940100]      <1>       mov    ah, [audio_user]
2274 00012E55 08E4                <1>       or     ah, ah
2275 00012E57 7418                <1>       jz     short sndc_init0
2276 00012E59 38E0                <1>       cmp    al, ah
2277 00012E5B 7419                <1>       je     short sndc_init1
2278                              <1>
2279                              <1> sndc_owner_error:
2280 00012E5D B80B000000          <1>       mov    eax, ERR_NOT_OWNER ; 'permission denied !' error
2281                              <1> sndc_perm_error:
2282 00012E62 A3[64030300]        <1>       mov    [u.r0], eax
2283 00012E67 A3[C8030300]        <1>       mov    [u.error], eax
2284 00012E6C E95BAAFFFF          <1>       jmp    error
2285                              <1> sndc_init0:
2286 00012E71 A2[39940100]        <1>       mov    [audio_user], al
2287                              <1> sndc_init1:
2288 00012E76 8915[3C940100]      <1>       mov    [audio_cb_addr], edx
2289 00012E7C 881D[3A940100]      <1>       mov    [audio_cb_mode], bl
2290 00012E82 880D[3B940100]      <1>       mov    [audio_srb], cl
2291                              <1>
2292                              <1>       ; 27/07/2020
2293                              <1>       ;mov  byte [audio_flag], 0  ; clear dma half buffer flag
2294                              <1>
2295                              <1>       ; 24/04/2017
2296 00012E88 803D[11940100]03    <1>       cmp    byte [audio_device], 3 ; VT8233 (VT8237R)
2297 00012E8F 7438                <1>       je     short sndc_init_9
2298                              <1>       ;ja    short soundc_respond_err ; temporary (28/05/2017)
2299 00012E91 803D[11940100]01    <1>       cmp    byte [audio_device], 1 ; SB 16
2300 00012E98 7510                <1>       jne    short sndc_init_13
2301 00012E9A BB[864C0100]        <1>       mov    ebx, sb16_int_handler
2302                              <1>       ; Note: 'SbInit' is at 'Start to Play' stage
2303                              <1>       ; 20/05/2017
2304 00012E9F 66C705[46940100]08- <1>       mov    word [audio_master_volume], 0808h ; 2/8
2304 00012EA7 08                  <1>
2305 00012EA8 EB3F                <1>       jmp    short sndc_init_10
2306                              <1> sndc_init_13:
2307                              <1>       ; 28/05/2017
2308 00012EAA 803D[11940100]02    <1>       cmp    byte [audio_device], 2 ; AC 97 (ICH)
2309 00012EB1 0F8562FEFFFF        <1>       jne    soundc_respond_err ; temporary (28/05/2017)
2310                              <1>
2311 00012EB7 E81F1F0000          <1>       call   ac97_codec_config
2312 00012EBC 0F8257FEFFFF        <1>       jc     soundc_respond_err ; codec error !
2313                              <1>
2314 00012EC2 BB[C24F0100]        <1>       mov    ebx, ac97_int_handler
2315 00012EC7 EB20                <1>       jmp    short sndc_init_10
2316                              <1>
2317                              <1> sndc_init_9:
2318                              <1>       ;call  reset_codec
2319                              <1>       ;; eax = 1
2320                              <1>       ;call  codec_io_w16 ; w32
2321 00012EC9 E8E6170000          <1>       call   init_codec ; 28/05/2017
2322 00012ECE 0F8245FEFFFF        <1>       jc     soundc_respond_err ; codec error !
2323                              <1>
2324 00012ED4 E80D1A0000          <1>       call   channel_reset
2325                              <1>
2326                              <1>       ; setup the Codec (actually mixer registers)
2327 00012ED9 E82D190000          <1>         call    codec_config  ; unmute codec, set rates.
2328 00012EDE 0F8235FEFFFF        <1>       jc     soundc_respond_err ; codec error !
2329                              <1>
2330 00012EE4 BB[78480100]        <1>       mov    ebx, vt8233_int_handler
2331                              <1> sndc_init_10:
2332                              <1>       ; 13/04/2017
2333 00012EE9 A0[13940100]        <1>       mov    al, [audio_intr] ; IRQ number
2334 00012EEE E82BFDFFFF          <1>       call   set_dev_IRQ_service
2335                              <1>
2336                              <1>       ; SETUP (audio) INTERRUPT CALLBACK SERVICE
2337 00012EF3 8A1D[13940100]      <1>       mov    bl, [audio_intr] ; IRQ number
2338 00012EF9 8A3D[3A940100]      <1>       mov    bh, [audio_cb_mode]
2339 00012EFF FEC7                <1>       inc    bh  ; 1 = Signal Response Byte method (fixed value)
2340                              <1>                  ; 2 = Callback service method
```

```
                                <1>                 ; 3 = Auto Increment S.R.B. method
00012F01 8A0D[3B940100]         <1>         mov     cl, [audio_srb]
00012F07 8B15[3C940100]         <1>         mov     edx, [audio_cb_addr]
00012F0D A0[39940100]           <1>         mov     al, [audio_user]
                                <1>         ; 14/04/2017
00012F12 E8E1040000             <1>         call    set_irq_callback_service
                                <1>         ; 16/04/2017
00012F17 A3[64030300]           <1>         mov     [u.r0], eax
                                <1>         ;jnc    sysret
00012F1C 7316                   <1>         jnc     short sndc_init2 ; 21/04/2017
                                <1>         ;
00012F1E A3[C8030300]           <1>         mov     dword [u.error], eax
                                <1>
00012F23 A0[13940100]           <1>         mov     al, [audio_intr] ; IRQ number
00012F28 31DB                   <1>         xor     ebx, ebx ; reset IRQ handler address
00012F2A E8EFFCFFFF             <1>         call    set_dev_IRQ_service
                                <1>
00012F2F E998A9FFFF             <1>         jmp     error
                                <1>
                                <1> sndc_init2:
                                <1>         ; 21/04/2017
00012F34 8B0D[2C940100]         <1>         mov     ecx, [audio_buff_size] ; audio buffer size
00012F3A D1E1                   <1>         shl     ecx, 1 ; *2
00012F3C A1[30940100]           <1>         mov     eax, [audio_dma_buff]
00012F41 21C0                   <1>         and     eax, eax
00012F43 7415                   <1>         jz      short sndc_init3
                                <1>
00012F45 8B15[34940100]         <1>         mov     edx, [audio_dmabuff_size] ; dma buffer size
00012F4B 39D1                   <1>         cmp     ecx, edx
00012F4D 744D                   <1>         je      short sndc_init5
                                <1>
00012F4F 87CA                   <1>         xchg    ecx, edx
00012F51 E82A37FFFF             <1>         call    deallocate_memory_block
00012F56 87D1                   <1>         xchg    edx, ecx
00012F58 31C0                   <1>         xor     eax, eax
                                <1> sndc_init3:
                                <1>         ; 12/05/2017
00012F5A 803D[11940100]01       <1>         cmp     byte [audio_device], 1 ; SB 16
00012F61 7515                   <1>         jne     short sndc_init4
00012F63 C705[30940100]-        <1>         mov     dword [audio_dma_buff], sb16_dma_buffer
00012F69 [00000200]             <1>
00012F6D C705[34940100]0000-    <1>         mov     dword [audio_dmabuff_size], 65536
00012F75 0100                   <1>
                                <1>         ;xor    eax, eax
                                <1>         ;mov    [u.r0], eax ; 0 = no error, successful
00012F77 C3                     <1>         retn
                                <1>
                                <1> sndc_init4:
                                <1>         ; EAX = Beginning address (physical)
                                <1>         ; EAX = 0 -> Allocate mem block from the 1st proper aperture
                                <1>         ; ECX = Number of bytes to be allocated (>0)
00012F78 E8F634FFFF             <1>         call    allocate_memory_block
00012F7D 0F828FFDFFFF           <1>         jc      sound_buff_error
                                <1>
                                <1>         ; set dma buffer address and size parameters
00012F83 A3[30940100]           <1>         mov     [audio_dma_buff], eax ; dma buffer address
00012F88 890D[34940100]         <1>         mov     [audio_dmabuff_size], ecx ; dma buffer size
                                <1> ;       ; EAX = Beginning (physical) addr of the allocated mem block
                                <1> ;       ; ECX = Num of allocated bytes (rounded up to page borders)
                                <1> ;       cmp     byte [audio_pci], 0 ; AC97 audio controller ?
                                <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:
00012F8E 803D[11940100]03       <1>         cmp     byte [audio_device], 3
                                <1>         ;jne    short sndc_init5
00012F95 7506                   <1>         jne     short sndc_init14 ; 28/05/2017
00012F97 E88B190000             <1>         call    set_vt8233_bdl
                                <1> sndc_init5:
                                <1>         ;sub    eax, eax ; 0
                                <1>         ;mov    [u.r0], eax ; 0 = no error, successful
00012F9C C3                     <1>         retn
                                <1> sndc_init14:
00012F9D E8521F0000             <1>         call    set_ac97_bdl
                                <1>         ;jmp    short sndc_init5
00012FA2 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
00012FA3 A0[11940100]           <1>       mov   al, [audio_device]
00012FA8 08C0                   <1>       or    al, al ; 0 ; pc speaker or invalid
00012FAA 0F843AF4FEFF           <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
00012FB0 833D[24940100]00       <1>       cmp   dword [audio_buffer], 0
00012FB7 0F8655FDFFFF           <1>       jna   sound_buff_error
00012FBD A0[B3030300]           <1>       mov   al, [u.uno]
00012FC2 3A05[39940100]         <1>       cmp   al, [audio_user]
00012FC8 0F858FFEFFFF           <1>       jne   sndc_owner_error
                                <1>
00012FCE 66890D[42940100]       <1>       mov   [audio_freq], cx ; sample frequency (Hertz)
00012FD5 88D8                   <1>       mov   al, bl
00012FD7 2401                   <1>       and   al, 1 ; mono/stereo (1= stereo)
00012FD9 FEC0                   <1>       inc   al ; channels
00012FDB A2[41940100]           <1>       mov   [audio_stmo], al ; sound channels (1 or 2)
00012FE0 B008                   <1>       mov   al, 8
00012FE2 F6C302                 <1>       test  bl, 2 ; bits per sample (1= 16 bit)
00012FE5 7402                   <1>       jz    short snd_play_bps
00012FE7 D0E0                   <1>       shl   al, 1
                                <1> snd_play_bps:
00012FE9 A2[40940100]           <1>       mov   [audio_bps], al
                                <1>
                                <1>       ; Transfer ring 3 (user's) audio buffer content to dma buffer
00012FEE 8B3D[30940100]         <1>       mov   edi, [audio_dma_buff] ; dma buffer (ring 0)
00012FF4 09FF                   <1>       or    edi, edi
00012FF6 0F8416FDFFFF           <1>       jz    sound_buff_error
                                <1>
                                <1>       ; 27/07/2020
                                <1>
00012FFC 8B35[28940100]         <1>       mov   esi, [audio_p_buffer] ; physical address (ring 3)
                                <1>       ;mov  ecx, [audio_buff_size] ; 15/05/2017
00013002 8B0D[34940100]         <1>       mov   ecx, [audio_dmabuff_size] ; 27/07/2020
                                <1>       ;or   ecx, ecx
                                <1>       ;jz   sound_buff_error
                                <1>       ; 28/07/2020
00013008 D1E9                   <1>       shr   ecx, 1  ; dma half buffer size
                                <1>
0001300A 8035[38940100]01       <1>       xor   byte [audio_flag], 1 ;  0 -> 1, 1 -> 0
00013011 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
00013013 01CF                   <1>       add   edi, ecx
                                <1>
                                <1> snd_play_0:
                                <1>       ;rep  movsb
00013015 C1E902                 <1>       shr   ecx, 2  ; convert byte count to dword count
00013018 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
0001301A A0[11940100]           <1>       mov   al, [audio_device]
0001301F 3C03                   <1>       cmp   al, 3 ; VT8233 (VT8237R)
00013021 7410                   <1>       je    short snd_play_1
00013023 3C01                   <1>       cmp   al, 1 ; Sound Blaster 16
00013025 7512                   <1>       jne   short snd_play_2  ; 28/05/2017
00013027 E8031B0000             <1>       call  SbInit_play
0001302C 0F82E7FCFFFF           <1>       jc    soundc_respond_err
00013032 C3                     <1>       retn
                                <1>
                                <1> snd_play_1:
00013033 E826190000             <1>       call  vt8233_start_play
00013038 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>
00013039 E8EA1E0000             <1>       call  ac97_start_play
0001303E 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
```

```
2549 0001303F E814030000          <1>       call   snd_dev_check
2550 00013044 7275                <1>       jc     short snd_nothing ; temporary.
2551 00013046 E81A030000          <1>       call   snd_buf_check
2552 0001304B 726E                <1>       jc     short snd_nothing ; temporary.
2553 0001304D A0[11940100]        <1>       mov    al, [audio_device]
2554 00013052 3C03                <1>       cmp    al, 3 ; VIA VT 8237R (vt8233)
2555 00013054 7409                <1>       je     short snd_pause_1
2556 00013056 3C01                <1>       cmp    al, 1 ; Sound Blaster 16
2557 00013058 750A                <1>       jne    short snd_pause_2 ; 28/05/2017
2558 0001305A E9AE1C0000          <1>       jmp    sb16_pause
2559                              <1> snd_pause_1:
2560 0001305F E9B3190000          <1>       jmp    vt8233_pause
2561                              <1> snd_pause_2:
2562                              <1>       ; 28/05/2017
2563                              <1>       ;cmp   al, 2 ; AC'97
2564                              <1>       ;jne   short snd_nothing ; temporary.
2565 00013064 E94D200000          <1>       jmp    ac97_pause
2566                              <1>
2567                              <1> sound_continue:
2568                              <1>       ; FUNCTION = 6
2569                              <1>       ; Continue to play
2570                              <1>       ; 28/05/2017
2571                              <1>       ; 22/04/2017
2572 00013069 E8EA020000          <1>       call   snd_dev_check
2573 0001306E 724B                <1>       jc     short snd_nothing ; temporary.
2574 00013070 E8F0020000          <1>       call   snd_buf_check
2575 00013075 7244                <1>       jc     short snd_nothing ; temporary.
2576 00013077 A0[11940100]        <1>       mov    al, [audio_device]
2577 0001307C 3C03                <1>       cmp    al, 3 ; VIA VT 8237R (vt8233)
2578 0001307E 7409                <1>       je     short snd_cont_1
2579 00013080 3C01                <1>       cmp    al, 1 ; Sound Blaster 16
2580 00013082 750A                <1>       jne    short snd_cont_2 ; 28/05/2017
2581 00013084 E9A71C0000          <1>       jmp    sb16_continue
2582                              <1> snd_cont_1:
2583 00013089 E93A190000          <1>       jmp    vt8233_play
2584                              <1> snd_cont_2:
2585                              <1>       ; 28/05/2017
2586                              <1>       ;cmp   al, 2 ; AC'97
2587                              <1>       ;jne   short snd_nothing ; temporary.
2588 0001308E E9EB1E0000          <1>       jmp    ac97_play
2589                              <1>
2590                              <1> sound_stop:
2591                              <1>       ; FUNCTION = 7
2592                              <1>       ; Stop playing
2593                              <1>       ; 28/05/2017
2594                              <1>       ; 24/05/2017
2595                              <1>       ; 21/04/2017, 22/04/2017, 24/04/2017
2596 00013093 E8C0020000          <1>       call   snd_dev_check
2597 00013098 7221                <1>       jc     short snd_nothing ; temporary.
2598                              <1>       ;call  snd_buf_check
2599 0001309A E8CF020000          <1>       call   snd_user_check ; 24/05/2017
2600 0001309F 721A                <1>       jc     short snd_nothing ; temporary.
2601                              <1>
2602 000130A1 A0[11940100]        <1>       mov    al, [audio_device]
2603 000130A6 3C03                <1>       cmp    al, 3 ; VIA VT 8237R (vt8233)
2604 000130A8 0F8470180000        <1>       je     vt8233_stop
2605                              <1>       ; 28/05/2017
2606                              <1>       ;ja    short snd_nothing
2607 000130AE 3C01                <1>       cmp    al, 1 ; Sound Blaster 16
2608 000130B0 0F849D1C0000        <1>       je     sb16_stop
2609                              <1>       ;cmp   al, 2
2610                              <1>       ;je    short ac97_stop
2611 000130B6 E9CD1F0000          <1>       jmp    ac97_stop ; temporary.
2612                              <1>       ;jmp   hda_stop
2613                              <1>
2614                              <1> snd_nothing:
2615                              <1>       ; 21/04/2017
2616 000130BB C3                  <1>       retn
2617                              <1>
2618                              <1> soundc_reset:
2619                              <1>       ; FUNCTION = 8
2620                              <1>       ; Reset Audio Controller
2621                              <1>       ; 28/05/2017
2622                              <1>       ; 22/04/2017
2623 000130BC E897020000          <1>       call   snd_dev_check
2624 000130C1 72F8                <1>       jc     snd_nothing ; temporary.
2625 000130C3 E89D020000          <1>       call   snd_buf_check
2626 000130C8 72F1                <1>       jc     snd_nothing ; temporary.
2627                              <1>
2628 000130CA A0[11940100]        <1>       mov    al, [audio_device]
2629 000130CF 3C03                <1>       cmp    al, 3 ; VIA VT 8237R (vt8233)
2630 000130D1 0F844C190000        <1>       je     vt8233_reset
2631 000130D7 77E2                <1>       ja     short snd_nothing ; temporary.
2632                              <1>       ;ja    hda_reset
2633 000130D9 3C01                <1>       cmp    al, 1 ; Sound Blaster 16
2634 000130DB 0F8526200000        <1>       jne    ac97_reset
2635 000130E1 E8BF1C0000          <1>       call   sb16_reset
2636 000130E6 0F822DFCFFFF        <1>       jc     soundc_respond_err
2637 000130EC C3                  <1>       retn
2638                              <1>
2639                              <1> soundc_cancel:
2640                              <1>       ; FUNCTION = 9
2641                              <1>       ; Cancel audio callback service
2642                              <1>       ; 22/04/2017
2643 000130ED A0[39940100]        <1>       mov    al, [audio_user]
2644 000130F2 3A05[B3030300]      <1>       cmp    al, [u.uno]
2645 000130F8 75C1                <1>       jne    short snd_nothing
2646                              <1>       ; RESET (audio) INTERRUPT CALLBACK SERVICE
2647 000130FA 8A1D[13940100]      <1>       mov    bl, [audio_intr] ; IRQ number
2648 00013100 A0[B3030300]        <1>       mov    al, [u.uno]
2649 00013105 28FF                <1>       sub    bh, bh ; 0 ; unlink IRQ from user service
2650 00013107 E8EC020000          <1>       call   set_irq_callback_service
2651 0001310C 0F8250FDFFFF        <1>       jc     sndc_perm_error ; 'permission denied' error
2652 00013112 C3                  <1>       retn
2653                              <1>
```

```
                            <1> sound_dalloc:
                            <1>       ; FUNCTION = 10
                            <1>       ; Deallocate (ring 3) audio buffer
                            <1>       ; 22/04/2017
00013113 A0[39940100]       <1>       mov   al, [audio_user]
00013118 3A05[B3030300]     <1>       cmp   al, [u.uno]
0001311E 759B               <1>       jne   short snd_nothing
00013120 8B1D[24940100]     <1>       mov   ebx, [audio_buffer]
                            <1>       ;or   ebx, ebx
                            <1>       ;jz   short snd_nothing
00013126 8B0D[2C940100]     <1>       mov   ecx, [audio_buff_size]
0001312C E89D36FFFF         <1>       call  deallocate_user_pages
00013131 31C0               <1>       xor   eax, eax
00013133 A3[24940100]       <1>       mov   [audio_buffer], eax ; 0
00013138 A2[39940100]       <1>       mov   [audio_user], al ; 0
0001313D 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>
0001313E 80FB80             <1>       cmp   bl, 80h
00013141 720E               <1>       jb    short snd_vol_1
00013143 0F8772FFFFFF       <1>       ja    snd_nothing ; temporary.
                            <1>       ; Set volume level for next play (BL>= 80h)
00013149 66890D[46940100]   <1>       mov   [audio_master_volume], cx
00013150 C3                 <1>       retn
                            <1> snd_vol_1:
                            <1>       ; set volume level immediate (BL< 80h)
00013151 80FB00             <1>       cmp   bl, 0
00013154 0F8761FFFFFF       <1>       ja    snd_nothing ; temporary.
                            <1>
0001315A E8F9010000         <1>       call  snd_dev_check
0001315F 0F8256FFFFFF       <1>       jc    snd_nothing ; temporary.
00013165 E8FB010000         <1>       call  snd_buf_check
0001316A 0F824BFFFFFF       <1>       jc    snd_nothing ; temporary.
                            <1>
00013170 A0[11940100]       <1>       mov   al, [audio_device]
00013175 3C03               <1>       cmp   al, 3 ; VIA VT 8237R (vt8233)
00013177 0F84BF180000       <1>       je    vt8233_volume
                            <1>       ; 28/05/2017
0001317D 0F8738FFFFFF       <1>       ja    snd_nothing ; temporary.
                            <1>       ;ja   hda_volume
                            <1>       ; Sound Blaster 16
00013183 3C01               <1>       cmp   al, 1 ; SB 16
00013185 0F844D1B0000       <1>       je    sb16_volume
0001318B E90A1E0000         <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
00013190 E8C3010000         <1>       call  snd_dev_check
00013195 0F8270FBFFFF       <1>       jc    soundc_dev_err ; temporary.
                            <1>       ;call  snd_buf_check
                            <1>       ;jc   sndc_owner_error ; temporary.
                            <1>
0001319B A0[11940100]       <1>       mov   al, [audio_device]
000131A0 3C03               <1>       cmp   al, 3 ; VIA VT 8237R (vt8233)
000131A2 7418               <1>       je    short snd_disable_1
000131A4 0F8711FFFFFF       <1>       ja    snd_nothing ; temporary.
000131AA 3C01               <1>       cmp   al, 1 ; Sound Blaster 16
000131AC 7507               <1>       jne   short snd_disable_0
000131AE E8A01B0000         <1>       call  sb16_stop
000131B3 EB0C               <1>       jmp   short snd_disable_2
                            <1> snd_disable_0:
000131B5 E8CE1E0000         <1>       call  ac97_stop
000131BA EB05               <1>       jmp   short snd_disable_2
                            <1> snd_disable_1:
000131BC E85D170000         <1>       call  vt8233_stop
                            <1> snd_disable_2:
000131C1 A0[13940100]       <1>       mov   al, [audio_intr]
000131C6 29DB               <1>       sub   ebx, ebx ; 0 = reset
000131C8 E851FAFFFF         <1>       call  set_dev_IRQ_service
                            <1>
                            <1>       ;mov  al, [audio_intr]
000131CD 28E4               <1>       sub   ah, ah ; 0 = reset
000131CF E8B5F6FFFF         <1>       call  set_hardware_int_vector
                            <1>
000131D4 31C0               <1>       xor   eax, eax
000131D6 A2[11940100]       <1>       mov   byte [audio_device], al
000131DB A2[13940100]       <1>       mov   byte [audio_intr], al
000131E0 8705[30940100]     <1>       xchg  eax, [audio_dma_buff]
                            <1>       ; 24/05/2017
                            <1>       ;or   eax, eax
                            <1>       ;jz   short snd_disable_3
                            <1>       ;cmp  eax, sb16_dma_buffer ; default DMA buffer
                            <1>       ;je   short snd_disable_3
000131E6 803D[10940100]00   <1>       cmp   byte [audio_pci], 0 ; AC97 audio controller ?
000131ED 7612               <1>       jna   short snd_disable_3
000131EF C605[10940100]00   <1>       mov   byte [audio_pci], 0
                            <1>       ;sub  ecx, ecx
                            <1>       ;xchg ecx, [audio_dmabuff_size]
000131F6 8B0D[34940100]     <1>       mov   ecx, [audio_dmabuff_size]
000131FC E87F34FFFF         <1>       call  deallocate_memory_block
                            <1> snd_disable_3:
00013201 C3                 <1>       retn
```

```
2759                                <1>
2760                                <1> sound_dma_map:
2761                                <1>       ; FUNCTION = 13
2762                                <1>       ; Map audio dma buff addr to user's buffer addr
2763                                <1>       ; 12/05/2017
2764 00013202 21C9                 <1>       and    ecx, ecx
2765 00013204 0F8408FBFFFF         <1>       jz     sound_buff_error
2766 0001320A 803D[11940100]01     <1>       cmp    byte [audio_device], 1
2767 00013211 7229                 <1>       jb     short snd_dma_map_1
2768                                <1> snd_dma_map_0:
2769 00013213 A1[30940100]         <1>       mov    eax, [audio_dma_buff]
2770 00013218 21C0                 <1>       and    eax, eax
2771 0001321A 7420                 <1>       jz     short snd_dma_map_1
2772                                <1>       ;
2773 0001321C 8A1D[39940100]       <1>       mov    bl, [audio_user]
2774 00013222 08DB                 <1>       or     bl, bl
2775 00013224 7416                 <1>       jz     short snd_dma_map_1
2776 00013226 3A1D[B3030300]       <1>       cmp    bl, [u.uno]
2777 0001322C 0F852BFCFFFF         <1>       jne    sndc_owner_error
2778                                <1>       ;
2779 00013232 8B1D[34940100]       <1>       mov    ebx, [audio_dmabuff_size]
2780 00013238 21DB                 <1>       and    ebx, ebx
2781 0001323A 750A                 <1>       jnz    short snd_dma_map_2
2782                                <1> snd_dma_map_1:
2783 0001323C B8[00000200]         <1>       mov    eax, sb16_dma_buffer
2784 00013241 BB00000100           <1>       mov    ebx, 65536
2785                                <1> snd_dma_map_2:
2786 00013246 81C1FF0F0000         <1>       add    ecx, PAGE_SIZE-1 ; 4095
2787 0001324C 6681E100F0           <1>       and    cx, ~PAGE_OFF ; not 4095
2788 00013251 39D9                 <1>       cmp    ecx, ebx
2789 00013253 0F87B9FAFFFF         <1>       ja     sound_buff_error
2790 00013259 50                   <1>       push   eax
2791 0001325A 89D3                 <1>       mov    ebx, edx
2792 0001325C C1E90C               <1>       shr    ecx, 12 ; byte count to page count
2793                                <1>       ; eax = physical address of (audio) dma buffer
2794                                <1>       ; ebx = virtual address of (audio) dma buffer (user's pgdir)
2795                                <1>       ; ecx = page count (>0)
2796 0001325F E88C34FFFF           <1>       call   direct_memory_access
2797 00013264 58                   <1>       pop    eax
2798 00013265 0F82A7FAFFFF         <1>       jc     sound_buff_error
2799 0001326B A3[64030300]         <1>       mov    [u.r0], eax
2800 00013270 C3                   <1>       retn
2801                                <1>
2802                                <1> soundc_info:
2803                                <1>       ; FUNCTION = 14
2804                                <1>       ; Get Audio Controller Info
2805                                <1>       ; 10/06/2017
2806                                <1>       ; 05/06/2017
2807 00013271 20DB                 <1>       and    bl, bl ; 0
2808 00013273 740A                 <1>       jz     short sndc_info_0
2809                                <1>       ; invalid parameter !
2810 00013275 B817000000           <1>       mov    eax, ERR_INV_PARAMETER ; 23
2811                                <1> ;sndc_inf_error:
2812                                <1> ;     mov    [u.r0], eax
2813                                <1> ;     mov    [u.error], eax
2814                                <1> ;     jmp    error
2815 0001327A E99FFAFFFF           <1>       jmp    sysaudio_err
2816                                <1>
2817                                <1> sndc_info_0:
2818 0001327F E8D4000000           <1>       call   snd_dev_check
2819 00013284 0F8281FAFFFF         <1>       jc     soundc_dev_err
2820                                <1>
2821 0001328A 8B1D[1C940100]       <1>       mov    ebx, [audio_vendor]
2822 00013290 8B0D[18940100]       <1>       mov    ecx, [audio_dev_id]
2823                                <1>       ;mov   al,  [audio_device]
2824 00013296 3C02                 <1>       cmp    al, 2 ; AC'97 (ICH)
2825 00013298 7513                 <1>       jne    short sndc_info_1
2826                                <1>       ; Intel AC97 (ICH) Audio Controller (=2)
2827 0001329A 668B15[4A940100]     <1>       mov    dx, [NABMBAR]
2828 000132A1 C1E210               <1>       shl    edx, 16
2829 000132A4 668B15[48940100]     <1>       mov    dx, [NAMBAR]
2830 000132AB EB07                 <1>       jmp    short sndc_info_2
2831                                <1> sndc_info_1:
2832                                <1>       ; 05/06/2017
2833                                <1>       ; Note: Intel HDA code (here) is not ready yet!
2834                                <1>       ; !!! SB16 or VT8233 (VT8237R) !!!
2835 000132AD 0FB715[16940100]     <1>       movzx  edx, word [audio_io_base]
2836                                <1> sndc_info_2:
2837 000132B4 88C4                 <1>       mov    ah, al ;  [audio_device]
2838 000132B6 A0[13940100]         <1>       mov    al, [audio_intr]
2839                                <1>
2840                                <1>       ; EAX = IRQ Number in AL
2841                                <1>       ;      Audio Device Number in AH
2842                                <1>       ; EBX = DEV/VENDOR ID
2843                                <1>       ;      (DDDDDDDDDDDDDDDDVVVVVVVVVVVVVVVV)
2844                                <1>       ; ECX = BUS/DEV/FN
2845                                <1>       ;      (00000000BBBBBBBBDDDDDFFF00000000)
2846                                <1>       ; EDX = NABMBAR/NAMBAR (for AC97)
2847                                <1>       ;      (Low word, DX = NAMBAR address)
2848                                <1>       ; EDX = Base IO Addr (DX) for SB16 & VT8233
2849                                <1>
2850                                <1>       ; 10/06/2017
2851 000132BB A3[64030300]         <1>       mov    [u.r0], eax
2852 000132C0 8B2D[60030300]       <1>       mov    ebp, [u.usp]
2853 000132C6 895D10               <1>       mov    [ebp+16], ebx  ; ebx
2854 000132C9 895514               <1>       mov    [ebp+20], edx  ; edx
2855 000132CC 894D18               <1>       mov    [ebp+24], ecx  ; ecx
2856                                <1>
2857 000132CF C3                   <1>       retn
2858                                <1>
2859                                <1> sound_data:
2860                                <1>       ; FUNCTION = 15
2861                                <1>       ; Get Current Sound data for graphics
2862                                <1>       ; 22/06/2017
2863                                <1>       ;
```

```
2864 000132D0 E883000000          <1>       call   snd_dev_check
2865 000132D5 0F8230FAFFFF        <1>       jc     soundc_dev_err ; Device not ready !
2866                              <1>
2867 000132DB 80FB00              <1>       cmp    bl, 0
2868 000132DE 760A                <1>       jna    short sound_data_0
2869                              <1>
2870                              <1>       ; Only PCM OUT buffer data is valid for now!
2871 000132E0 B817000000          <1>       mov    eax, ERR_INV_PARAMETER  ; 23
2872 000132E5 E934FAFFFF          <1>       jmp    sysaudio_err
2873                              <1>
2874                              <1> sound_data_0:
2875 000132EA A1[30940100]        <1>       mov    eax, [audio_dma_buff]
2876 000132EF 09C0                <1>       or     eax, eax
2877 000132F1 0F841BFAFFFF        <1>       jz     sound_buff_error
2878                              <1>
2879 000132F7 803D[11940100]04    <1>       cmp    byte [audio_device], 4 ; Intel HDA
2880 000132FE 744F                <1>       je     short sound_data_4 ; temporary ! (22/06/2017)
2881                              <1>
2882 00013300 21C9                <1>       and    ecx, ecx
2883                              <1>       ;jnz   short sound_data_1 ; sample tranfer
2884                              <1>
2885                              <1>       ; Return only DMA Buffer pointer/offset...
2886                              <1>       ; (If DMA Buffer has been mapped to user's
2887                              <1>       ;  memory space; program can get graphics
2888                              <1>       ;  data by using only this pointer value.)
2889                              <1>
2890                              <1>       ;call  get_dma_buffer_offset
2891                              <1>       ;; eax = DMA buffer offset
2892                              <1>       ;;     (!not half buffer offset!)
2893                              <1>       ;mov   [u.r0], eax
2894                              <1>       ;retn
2895                              <1>
2896 00013302 0F84771F0000        <1>       jz     get_dma_buffer_offset
2897                              <1>
2898                              <1> sound_data_1:
2899                              <1>       ;mov   eax, [audio_dmabuff_size]
2900                              <1>       ;shr   eax, 1 ; half buffer size
2901                              <1>       ;cmp   ecx, eax
2902                              <1>       ;ja    short sound_buff_error
2903                              <1>
2904 00013308 3B0D[34940100]      <1>       cmp    ecx, [audio_dmabuff_size]
2905 0001330E 0F87FEF9FFFF        <1>       ja     sound_buff_error
2906                              <1>
2907 00013314 89D0                <1>       mov    eax, edx
2908 00013316 25FF0F0000          <1>       and    eax, PAGE_OFF ; 4095 (0FFFh)
2909 0001331B 81F900100000        <1>       cmp    ecx, 4096
2910 00013321 7605                <1>       jna    short sound_data_2
2911 00013323 B900100000          <1>       mov    ecx, 4096 ; max. 1 page
2912                              <1> sound_data_2:
2913 00013328 01C8                <1>       add    eax, ecx
2914 0001332A 3D00100000          <1>       cmp    eax, 4096
2915 0001332F 7606                <1>       jna    short sound_data_3
2916 00013331 6625FF0F            <1>       and    ax, PAGE_OFF ; 4095 (0FFFh)
2917 00013335 29C1                <1>       sub    ecx, eax
2918                              <1>       ; here, ECX has been adjusted to fit
2919                              <1>       ;      in page border..  (<= 4096, >0)
2920                              <1> sound_data_3:
2921 00013337 51                  <1>       push   ecx
2922 00013338 52                  <1>       push   edx
2923 00013339 89D3                <1>       mov    ebx, edx
2924 0001333B E89E2FFFFF          <1>       call   get_physical_addr
2925 00013340 5A                  <1>       pop    edx
2926 00013341 59                  <1>       pop    ecx
2927 00013342 0F82CAF9FFFF        <1>       jc     sound_buff_error
2928                              <1>
2929                              <1>       ; eax = physical address of user's buffer
2930 00013348 89C3                <1>       mov    ebx, eax
2931                              <1>       ; ecx = byte (transfer) count
2932                              <1>       ;call  get_current_sound_data
2933                              <1>       ;retn
2934 0001334A E98D1E0000          <1>       jmp    get_current_sound_data
2935                              <1>
2936                              <1> sound_data_4:
2937                              <1>       ; Intel HDA code is not ready yet !
2938                              <1>       ; 22/06/2017
2939 0001334F 31C0                <1>       xor    eax, eax
2940 00013351 48                  <1>       dec    eax
2941 00013352 A3[64030300]        <1>       mov    [u.r0], eax ; 0FFFFFFFFh
2942 00013357 C3                  <1>       retn
2943                              <1>
2944                              <1> snd_dev_check:
2945                              <1>       ; 10/06/2017
2946                              <1>       ; 05/06/2017
2947                              <1>       ; 24/05/2017
2948                              <1>       ; 22/04/2017
2949                              <1>       ; 21/04/2017
2950                              <1>       ; ... device check at first
2951 00013358 A0[11940100]        <1>       mov    al, [audio_device]
2952 0001335D 3C01                <1>       cmp    al, 1 ; SB 16
2953 0001335F 7203                <1>       jb     short snd_dev_chk_retn ; error !
2954                              <1>       ;cmp   al, 4 ; Intel HDA
2955                              <1>       ;ja    short snd_dbchk_stc ; invalid !
2956                              <1>       ; 10/06/2017
2957 00013361 3C05                <1>       cmp    al, 5
2958 00013363 F5                  <1>       cmc
2959                              <1> snd_dev_chk_retn:
2960 00013364 C3                  <1>       retn
2961                              <1>
2962                              <1> snd_buf_check:
2963                              <1>       ; 10/06/2017
2964                              <1>       ; 22/04/2017
2965                              <1>       ; 21/04/2017
2966                              <1>       ; ... buffer & (buffer) owner check at second
2967 00013365 833D[24940100]00    <1>       cmp    dword [audio_buffer], 0
2968 0001336C 760D                <1>       jna    short snd_dbchk_stc
```

```
                                <1> snd_user_check:
0001336E A0[B3030300]           <1>       mov    al, [u.uno]
00013373 3A05[39940100]         <1>       cmp    al, [audio_user]
                                <1>       ;jne   short snd_dbchk_stc
                                <1>       ;retn
00013379 74E9                   <1>       je     short snd_dev_chk_retn
                                <1>
                                <1> snd_dbchk_stc:
0001337B F9                     <1>       stc
0001337C C3                     <1>       retn
                                <1>
                                <1> sound_update:
                                <1>       ; FUNCTION = 16
                                <1>       ; bl =
                                <1>       ;    0 = automatic (sequental) update (with flag switch!)
                                <1>       ;    1 = update dma half buffer 1 (without flag switch!)
                                <1>       ;    2 = update dma half buffer 2 (without flag switch!)
                                <1>       ;  FFh = get current flag value
                                <1>       ;       0 = dma half buffer 1 (will be played next)
                                <1>       ;       1 = dma half buffer 2 (will be played next)
                                <1>
                                <1>       ; 10/10/2017
                                <1>
                                <1>       ; ... device check at first
0001337D A0[11940100]           <1>       mov    al, [audio_device]
00013382 08C0                   <1>       or     al, al ; 0 ; pc speaker or invalid
00013384 0F8481F9FFFF           <1>       jz     soundc_dev_err
                                <1>
                                <1>       ; ... buffer & (buffer) owner check at second
0001338A 833D[24940100]00       <1>       cmp    dword [audio_buffer], 0
00013391 0F867BF9FFFF           <1>       jna    sound_buff_error
00013397 A0[B3030300]           <1>       mov    al, [u.uno]
0001339C 3A05[39940100]         <1>       cmp    al, [audio_user]
000133A2 0F85B5FAFFFF           <1>       jne    sndc_owner_error
                                <1>
                                <1>       ; Transfer ring 3 (user's) audio buffer content to dma buffer
000133A8 8B3D[30940100]         <1>       mov    edi, [audio_dma_buff] ; dma buffer (ring 0)
000133AE 09FF                   <1>       or     edi, edi
000133B0 0F845CF9FFFF           <1>       jz     sound_buff_error
000133B6 8B35[28940100]         <1>       mov    esi, [audio_p_buffer] ; physical address (ring 3)
000133BC 8B0D[2C940100]         <1>       mov    ecx, [audio_buff_size]
                                <1>
                                <1>       ;movzx eax, byte [audio_flag]
000133C2 A0[38940100]           <1>       mov    al, [audio_flag]
                                <1>
000133C7 FEC3                   <1>       inc    bl
000133C9 7427                   <1>       jz     short snd_update_3 ; bl = 0FFh
000133CB FECB                   <1>       dec    bl
000133CD 7411                   <1>       jz     short snd_update_0 ; bl = 0
                                <1>
000133CF 80FB02                 <1>       cmp    bl, 2
000133D2 7417                   <1>       je     short snd_update_1 ; dma half buffer 2
000133D4 7217                   <1>       jb     short snd_update_2 ; dma half buffer 1
                                <1>
                                <1>       ; invalid parameter !
000133D6 B817000000             <1>       mov    eax, ERR_INV_PARAMETER ; 23
                                <1> ;     mov    [u.r0], eax
                                <1> ;     mov    [u.error], eax
                                <1> ;     jmp    error
000133DB E93EF9FFFF             <1>       jmp    sysaudio_err
                                <1>
                                <1> snd_update_0:
000133E0 8035[38940100]01       <1>       xor    byte [audio_flag], 1 ; update flag !!!
000133E7 3C01                   <1>       cmp    al, 1
000133E9 7202                   <1>       jb     short snd_update_2 ; dma half buffer 1
                                <1> snd_update_1:
                                <1>       ; dma half buffer 2
000133EB 01CF                   <1>       add    edi, ecx
                                <1> snd_update_2:
                                <1>       ;rep   movsb
000133ED C1E902                 <1>       shr    ecx, 2
000133F0 F3A5                   <1>       rep    movsd
                                <1> snd_update_3:
000133F2 A3[64030300]           <1>       mov    [u.r0], eax
                                <1>
000133F7 C3                     <1>       retn
                                <1>
                                <1> set_irq_callback_service:
                                <1>       ; 03/08/2020
                                <1>       ; 10/06/2017
                                <1>       ; 12/05/2017
                                <1>       ; 24/04/2017
                                <1>       ; 22/04/2017
                                <1>       ; caller: 'syscalbac' or 'sysaudio' or ...
                                <1>       ; 13/04/2017, 14/04/2017, 17/04/2017
                                <1>       ; 24/02/2017, 26/02/2017, 28/02/2017
                                <1>       ; 21/02/2017 - TRDOS 386 (TRDOS v2.0)
                                <1>       ;
                                <1>       ; Link or unlink IRQ callback service to/from user (ring 3)
                                <1>       ;
                                <1>       ; INPUT ->
                                <1>       ;     If AL = 0, the caller is 'syscalbac';
                                <1>       ;        otherwise, the caller is 'sysaudio' or ...
                                <1>       ;        (AL = user number)
                                <1>       ;
                                <1>       ;     BL = IRQ number (Hardware interrupt request number)
                                <1>       ;          (0 t0 15 but IRQ 0,1,2,6,8,14,15 are prohibited)
                                <1>       ;          IRQ numbers 3,4,5,7,9,10,11,12,13 are valid
                                <1>       ;          (numbers >15 are invalid)
                                <1>       ;
                                <1>       ;     BH = 0 = Unlink IRQ (in BL) from user (ring 3) service
                                <1>       ;          1 = Link IRQ by using Signal Response Byte method
                                <1>       ;          2 = Link IRQ by using Callback service method
                                <1>       ;          3 = Link IRQ by using Auto Increment S.R.B. method
                                <1>       ;         >3 = invalid
```

```
3074                                 <1>       ;              (syscallback version will return to user)
3075                                 <1>       ;
3076                                 <1>       ;      CL = Signal Return/Response Byte value
3077                                 <1>       ;
3078                                 <1>       ;      If BH = 3, kernel will put a counter value ; 03/08/2020
3079                                 <1>       ;                (into the S.R.B. addr)
3080                                 <1>       ;               between 0 to 255. (start value = CL+1)
3081                                 <1>       ;
3082                                 <1>       ;      NOTE: counter value, for example: even and odd numbers
3083                                 <1>       ;            may be used for -audio- DMA buffer switch
3084                                 <1>       ;            within double buffer method, etc.
3085                                 <1>       ;
3086                                 <1>       ;      EDX = Signal return (Response) byte address
3087                                 <1>       ;                            - or -
3088                                 <1>       ;            Interrupt/Callback service/routine address
3089                                 <1>       ;
3090                                 <1>       ;            (virtual address in user's memory space)
3091                                 <1>       ;
3092                                 <1>       ; OUTPUT ->
3093                                 <1>       ;      CF = 0 & EAX = 0 -> Successful setting
3094                                 <1>       ;      CF = 1 & EAX > 0 -> IRQ is prohibited or locked
3095                                 <1>       ;                    by another process
3096                                 <1>       ;              eax = ERR_PERM_DENIED -> prohibited or locked
3097                                 <1>       ;              eax = ERR_INV_PARAMETER ->
3098                                 <1>       ;                    invalid parameter/option or bad address
3099                                 <1>       ;
3100                                 <1>       ; TRDOS 386 - IRQ CALLBACK structures (parameters):
3101                                 <1>       ;
3102                                 <1>       ;         [u.irqlock] = 1 word, IRQ flags (0-15) that indicates
3103                                 <1>       ;                    which IRQs are locked by (that) user.
3104                                 <1>       ;                     Lock and unlock (by user) will change
3105                                 <1>       ;                    these flags or 'terminate process' (sysexit)
3106                                 <1>       ;                    will clear these flags and unlock those IRQs.
3107                                 <1>       ;
3108                                 <1>       ;                          Bit 0 is for IRQ 0 and Bit 15 is for IRQ 15
3109                                 <1>       ;
3110                                 <1>       ;         IRQ(x).owner      : 1 byte, user, [u.uno], 0 = free (unlocked)
3111                                 <1>       ;
3112                                 <1>       ;         IRQ(x).method : 1 byte for callback method & status
3113                                 <1>       ;                             0 = Signal Response Byte method
3114                                 <1>       ;                             1 = Callback service method
3115                                 <1>       ;                             >1 = invalid for current 'syscalback'.
3116                                 <1>       ;                           or(+) 80h = IRQ is in use by system (ring 0)
3117                                 <1>       ;                                       function (audio etc.) or
3118                                 <1>       ;                                  a device driver.
3119                                 <1>       ;                           (system function will ignore the lock/owner)
3120                                 <1>       ;
3121                                 <1>       ;         IRQ(x).srb: 1 byte, Signal Return/Response byte value
3122                                 <1>       ;                             (a fixed value by user or a counter value
3123                                 <1>       ;                            from 0 to 255, which is increased by every
3124                                 <1>       ;                            interrupt just before putting it into
3125                                 <1>       ;                            the Signal Response byte address
3126                                 <1>       ;                            (This is not used in callback serv method)
3127                                 <1>       ;
3128                                 <1>       ;         IRQ(x).addr       : 1 dword
3129                                 <1>       ;                             Signal Response Byte address (physical)
3130                                 <1>       ;                                        -or-
3131                                 <1>       ;                             Callback service address (virtual)
3132                                 <1>       ;
3133                                 <1>       ;         IRQ(x).dev: 1 byte
3134                                 <1>       ;                             0 = Default device or kernel function
3135                                 <1>       ;                                        -or-
3136                                 <1>       ;                             1-255 = Assigned device driver number
3137                                 <1>       ;
3138                                 <1>       ;         (x) = 3,4,5,7,9,10,11,12,13
3139                                 <1>       ;
3140                                 <1>
3141 000133F8 80FB0F                <1>       cmp    bl, 15
3142 000133FB 7729                  <1>       ja     short scbs_2
3143                                 <1>
3144 000133FD 80FF03                <1>       cmp    bh, 3
3145 00013400 7724                  <1>       ja     short scbs_2  ; invalid parameter
3146                                 <1>
3147 00013402 0FB6FB                <1>       movzx  edi, bl ; save IRQ number
3148                                 <1>
3149                                 <1>              ;  IRQ 0,1,2,6,8,14,15 are prohibited
3150                                 <1>       ;IRQenum: ; 'trdosk9.s'
3151                                 <1>       ;      db  0,0,0,1,2,3,0,4,0,5,6,7,8,9,0,0
3152                                 <1>
3153 00013405 0FB6B7[58410100]      <1>       movzx  esi, byte [edi+IRQenum] ; IRQ availability
3154                                 <1>                                      ; enumeration/index
3155                                 <1>       ;dec   esi
3156 0001340C 664E                  <1>       dec    si
3157 0001340E 780F                  <1>       js     short scbs_1 ;  0 -> 0FFFFh
3158                                 <1>
3159                                 <1>       ; ESI = IRQ callback parameters index number (0 to 8)
3160                                 <1>
3161 00013410 08FF                  <1>       or     bh, bh
3162 00013412 7419                  <1>       jz     short scbs_4 ; unlink the IRQ (in BL)
3163                                 <1>
3164 00013414 FECF                  <1>       dec    bh
3165                                 <1>       ; bh = method (0 = signal response byte, 1 = callback)
3166                                 <1>       ;            (2 = auto increment of signal response byte)
3167                                 <1>
3168 00013416 80BE[C2930100]00      <1>       cmp    byte [esi+IRQ.owner], 0 ; locked ?
3169 0001341D 7637                  <1>       jna    short scbs_6 ; no... OK...
3170                                 <1>
3171                                 <1> scbs_1:
3172                                 <1>       ; permission denied (prohibited IRQ)
3173 0001341F B80B000000            <1>       mov    eax, ERR_PERM_DENIED
3174 00013424 F9                    <1>       stc
3175 00013425 C3                    <1>       retn
3176                                 <1> scbs_2:
3177 00013426 F9                    <1>       stc
3178                                 <1> scbs_3:
```

```
3179 00013427 B817000000          <1>       mov    eax, ERR_INV_PARAMETER
3180 0001342C C3                  <1>       retn
3181                              <1>
3182                              <1> scbs_4: ; unlink the requested IRQ (if it belongs to current user)
3183                              <1>       ; 10/06/2017
3184                              <1>       ; 22/04/2017
3185                              <1>       ; 14/04/2017
3186                              <1>       ; If AL = 0 -> The caller is 'syscalbac'
3187 0001342D 8AA6[C2930100]      <1>       mov    ah, [esi+IRQ.owner]
3188 00013433 3A25[B3030300]      <1>       cmp    ah, [u.uno]
3189 00013439 75E4                <1>       jne    short scbs_1
3190                              <1>
3191 0001343B FE0D[D6030300]      <1>       dec    byte [u.irqc] ; decrease IRQ count (in use)
3192                              <1>
3193                              <1>       ;sub   ah, ah
3194                              <1>       ;mov   [esi+IRQ.owner], ah ; 0 ; free !!!
3195                              <1>       ;and   byte [esi+IRQ.method], 80h
3196                              <1>       ;mov   [esi+IRQ.srb], ah ; 0
3197                              <1>       ;mov   [esi+IRQ.dev], ah ; 0
3198                              <1>       ;mov   dword [esi+IRQ.addr], 0
3199                              <1>       ;mov   dword [u.r0], 0
3200                              <1>
3201                              <1>       ;mov   byte [esi+IRQ.owner], 0
3202                              <1>
3203                              <1>       ; 22/04/2017
3204 00013441 29C0                <1>       sub    eax, eax
3205 00013443 8886[C2930100]      <1>       mov    [esi+IRQ.owner], al ; 0
3206                              <1>       ; 10/06/2017
3207 00013449 8686[D4930100]      <1>       xchg   al, [esi+IRQ.method]
3208 0001344F 2480                <1>       and    al, 80h
3209 00013451 745E                <1>       jz     short scbs_12
3210                              <1>       ; Audio device must be disabled -later- ! ([IRQ.medhod] = 80h)
3211                              <1>
3212                              <1> ;     cmp    byte [esi+IRQ.method], 80h ; device drv or kernel extension ?
3213                              <1> ;     jb     short scbs_12 ; bh = 0 reset to default IRQ handler
3214                              <1> ;
3215                              <1> ;     and    al, al
3216                              <1> ;     jz     short  scbs_5  ; the caller is 'syscalbac'
3217                              <1> ;     ; The caller is 'sysaudio' or ...
3218 00013453 30C0                <1>       xor    al, al
3219                              <1> ;     mov    [esi+IRQ.method], al ; 0 ; reset kernel extension flag
3220                              <1> ;scbs_5:
3221                              <1> ;     sub    ah, ah
3222                              <1>       ;mov   [u.r0], eax ; 0
3223 00013455 C3                  <1>       retn
3224                              <1>
3225                              <1> scbs_6:
3226                              <1>       ; 14/04/2017
3227 00013456 20C0                <1>       and    al, al
3228 00013458 7405                <1>       jz     short scbs_7  ; the caller is 'syscalbac'
3229                              <1>       ; AL = user number ([u.uno] or [audio.user] or ...)
3230                              <1>       ; The caller is 'sysaudio' or ...
3231                              <1>       ;
3232                              <1>       ; bh = method (0 = signal response byte, 1 = callback)
3233                              <1>       ;            (2 = auto increment of signal response byte)
3234                              <1>
3235 0001345A 80CF80              <1>       or     bh, 80h             ; Kernel extension flag !
3236 0001345D EB0A                <1>       jmp    short scbs_8
3237                              <1> scbs_7:
3238 0001345F 8A86[D4930100]      <1>       mov    al, [esi+IRQ.method] ; >= 80h = kernel is using this IRQ
3239 00013465 2480                <1>       and    al, 80h ; use only bit 7 (kernel function flag)
3240 00013467 08C7                <1>       or     bh, al        ; method
3241                              <1>                             ; 0 = signal response byte, 1 = callback
3242                              <1>                             ; 2 = auto increment of s.r.b.
3243                              <1> scbs_8:
3244 00013469 A0[B3030300]        <1>       mov    al, [u.uno] ; user (process) number (1 to 16)
3245 0001346E 8886[C2930100]      <1>       mov    [esi+IRQ.owner], al ; lock the IRQ for user
3246 00013474 88BE[D4930100]      <1>       mov    [esi+IRQ.method], bh
3247                              <1>
3248                              <1> ;     test   bh, 1
3249                              <1> ;     jnz    short scbs_9  ; Callback method, CX will not be used
3250                              <1> ;
3251                              <1> ;     test   bh, 2         ; use auto increment (counter) method
3252                              <1> ;     jz     short scbs_10 ; (count can be used for buffer switch)
3253                              <1> ;scbs_9:
3254                              <1> ;     xor    ecx, ecx ; 0
3255                              <1> scbs_10:
3256                              <1>       ;mov   [esi+IRQ.method], bh
3257 0001347A 888E[DD930100]      <1>       mov    [esi+IRQ.srb], cl
3258 00013480 C686[CB930100]00    <1>       mov    byte [esi+IRQ.dev], 0 ; device number is always 0
3259                              <1>                             ; for this system call
3260                              <1>       ;test  bh, 1
3261 00013487 80E701              <1>       and    bh, 1 ; 17/04/2017
3262 0001348A 7513                <1>       jnz    short scbs_11 ; callback method, use virtual address
3263                              <1>
3264 0001348C 53                  <1>       push   ebx ; IRQ number (in BL)
3265 0001348D 89D3                <1>       mov    ebx, edx
3266                              <1>       ; ebx = virtual address
3267                              <1>       ; [u.pgdir] = page directory's physical address
3268 0001348F FE05[62930100]      <1>       inc     byte [no_page_swap] ; 1
3269                              <1>                   ; Do not add this page to swap queue
3270                              <1>                   ; and remove it from swap queue if it is
3271                              <1>                   ; on the queue.
3272 00013495 E8442EFFFF          <1>       call   get_physical_addr
3273 0001349A 5B                  <1>       pop    ebx
3274 0001349B 728A                <1>       jc     scbs_3 ; invalid address !
3275                              <1>       ; eax = physical address of the virtual address in user's space
3276 0001349D 89C2                <1>       mov    edx, eax
3277                              <1> scbs_11:
3278 0001349F 66C1E602            <1>       shl    si, 2        ; byte (index) to dword (offset)
3279 000134A3 8996[E6930100]      <1>       mov    [esi+IRQ.addr], edx
3280                              <1>
3281 000134A9 FE05[D6030300]      <1>       inc    byte [u.irqc]; increase IRQ (in use) count
3282                              <1>
3283 000134AF FEC7                <1>       inc     bh ; 17/04/2017
```

```
3284                            <1>       ; bh > 0 -> set to requested IRQ handler (IRQ_u_list)
3285                            <1> scbs_12:
3286 000134B1 88D8              <1>       mov    al, bl ; IRQ number
3287 000134B3 88FC              <1>       mov    ah, bh ; 0 = reset, >0 = set
3288 000134B5 E8CFF3FFFF        <1>       call   set_hardware_int_vector
3289                            <1>
3290 000134BA 31C0              <1>       xor    eax, eax
3291                            <1>       ;mov   [u.r0], eax ; 0
3292                            <1>
3293 000134BC C3                <1>       retn   ; return with success (cf=0, eax=0)
3294                            <1>
3295                            <1>
3296                            <1> sysdma: ; DMA FUNCTIONS
3297                            <1>       ; 02/09/2017
3298                            <1>       ; 28/08/2017
3299                            <1>       ; 20/08/2017 - TRDOS 386 (TRDOS v2.0)
3300                            <1>       ;
3301                            <1>       ; Inputs:
3302                            <1>       ;      BH = 0 -> Allocate DMA buffer
3303                            <1>       ;         BL = 0 -> Use the system's default DMA
3304                            <1>       ;                   (SB16) Buffer
3305                            <1>       ;              Buffer Size (max.) = 65536 bytes
3306                            <1>       ;         BL > 0 -> Allocate (a new) DMA buffer
3307                            <1>       ;         ECX = DMA Buffer Size in bytes (<=128KB)
3308                            <1>       ;         EDX = Virtual Address of DMA buffer
3309                            <1>       ;
3310                            <1>       ;      BH = 1 -> Initialize (Start) DMA service
3311                            <1>       ;           BL, bit 0 to 3 = Channel Number (0 to 7)
3312                            <1>       ;           BL, bit 7 = Auto Initialized Mode
3313                            <1>       ;                    (If bit 7 is set)
3314                            <1>       ;              bit 6 = Record (read) mode (0= playback)
3315                            <1>       ;           ECX = byte count (0 = use dma buffer size)
3316                            <1>       ;           EDX = physical buffer address
3317                            <1>       ;                 (0 = use dma buffer -start- address)
3318                            <1>       ;
3319                            <1>       ;      BH = 2 -> Get Current DMA Buffer Offset
3320                            <1>       ;           BL = DMA channel number
3321                            <1>       ;
3322                            <1>       ;      BH = 3 -> Get Current DMA count down value
3323                            <1>       ;           BL = DMA channel number (0 tO 7)
3324                            <1>       ;
3325                            <1>       ;      BH = 4 -> Get Current DMA channel (in progress)
3326                            <1>       ;
3327                            <1>       ;      BH = 5 -> Get System's Default DMA Buffer Address
3328                            <1>       ;
3329                            <1>       ;      BH = 6 -> Get Current DMA Buffer Address
3330                            <1>       ;
3331                            <1>       ;      BH = 7 -> Stop DMA service
3332                            <1>       ;
3333                            <1>       ;
3334                            <1>       ; Outputs:
3335                            <1>       ;
3336                            <1>       ;      For BH = 0 ; Allocate DMA buffer
3337                            <1>       ;          EAX = Physical address of DMA buffer
3338                            <1>       ;          ECX = Allocated buffer size in bytes
3339                            <1>       ;               - page count * 4096 -
3340                            <1>       ;              (may be bigger than requested)
3341                            <1>       ;          If BL input > 0,
3342                            <1>       ;             'sysalloc:' system call will be used with
3343                            <1>       ;             EBX (for 'sysalloc') = EDX (for 'sysdma')
3344                            <1>       ;             ECX is same, byte count (buffer size)
3345                            <1>       ;             EDX = 1024*1024*16 ; 16 MB upper limit
3346                            <1>       ;          If BL input = 0,
3347                            <1>       ;             Default DMA buffer (SB16 buffer) will be
3348                            <1>       ;             checked and if it is free, it's address
3349                            <1>       ;             will be returned in EAX and it's size
3350                            <1>       ;             will be returned in ECX (as 65536)
3351                            <1>       ;
3352                            <1>       ;          If CF = 1, error code is in EAX
3353                            <1>       ;             EAX = -1 ; DMA buffer allocation error!
3354                            <1>       ;             EAX = 11 ; 'Permission Denied' error !
3355                            <1>       ;
3356                            <1>       ;               Note: 'sysalloc' error return method
3357                            <1>       ;                     will be applied if BL input > 0 !
3358                            <1>       ;
3359                            <1>       ;       For BH = 1 ; Initialize (Start) DMA
3360                            <1>       ;           EAX = 0 (Successful)
3361                            <1>       ;           If CF = 1, error code is in EAX
3362                            <1>       ;
3363                            <1>       ;       For BH = 2 ; Get Current DMA Buffer Offset
3364                            <1>       ;           EAX = DMA Buffer Offset (in bytes)
3365                            <1>       ;           ;
3366                            <1>       ;            AX = DMA buffer offset
3367                            <1>       ;           EAX bits 16 to 23 = Page register value
3368                            <1>       ;
3369                            <1>       ;       For BH = 3 ; Get Current DMA count down value
3370                            <1>       ;           EAX = Count down value (remain bytes)
3371                            <1>       ;
3372                            <1>       ;       For BH = 4 ; Get Current DMA channel (in progress)
3373                            <1>       ;           EAX = DMA channel number (0 to 7)
3374                            <1>       ;             AH = 0 if the owner is the caller process
3375                            <1>       ;             AH > 0 if the dma channel is in use by
3376                            <1>       ;                    another user/process
3377                            <1>       ;           EAX = -1 (0FFFFFFFFh)
3378                            <1>       ;                if DMA service is not in use
3379                            <1>       ;                (stopped or not initialized/started)
3380                            <1>       ;
3381                            <1>       ;       For BH = 5 ; Get System's Default DMA Buff Addr
3382                            <1>       ;           EAX = Default DMA Buffer Address (Physical)
3383                            <1>       ;             = offset 'sb16_dma_buffer:'
3384                            <1>       ;           ECX = Buffer size
3385                            <1>       ;             = 65536
3386                            <1>       ;
3387                            <1>       ;       For BH = 6 ; Get Current DMA Buffer Address
3388                            <1>       ;           EAX = Current DMA buffer address (Physical)
```

```
3389                                    <1>       ;           ECX = Current DMA buffer size (setting value)
3390                                    <1>       ;           Note: These values are for current dma channel
3391                                    <1>       ;               settings for the user/process
3392                                    <1>       ;           ** For now (for current TRDOS 386 version)
3393                                    <1>       ;             only one user/process can use only one
3394                                    <1>       ;             dma channel & one dma buffer at same time
3395                                    <1>       ;             (no multi tasking on DMA service) !!! **
3396                                    <1>       ;           (Once, current DMA user must stop it's own DMA
3397                                    <1>       ;            DMA service, than another user/program
3398                                    <1>       ;            can use DMA service with same dma channel
3399                                    <1>       ;            or with another DMA channel.)
3400                                    <1>       ;
3401                                    <1>       ;       For BH = 7 ; Stop DMA service (for current user
3402                                    <1>       ;           and current DMA channel)
3403                                    <1>       ;           EAX = 0 ; successful
3404                                    <1>       ;           CF = 1 & EAX > 0 (= -1) -> Error
3405                                    <1>
3406 000134BD 80FF07                  <1>       cmp    bh, 7
3407 000134C0 7612                    <1>       jna    short sysdma_0
3408                                    <1>
3409                                    <1> sysdma_err:
3410 000134C2 31C0                    <1>       xor    eax, eax
3411 000134C4 48                      <1>       dec    eax ; -1
3412                                    <1> sysdma_perm_err:
3413 000134C5 A3[64030300]            <1>       mov    [u.r0], eax
3414 000134CA A3[C8030300]            <1>       mov    [u.error], eax ; DMA service error !
3415 000134CF E9F8A3FFFF              <1>       jmp    error
3416                                    <1>
3417                                    <1> sysdma_0:
3418 000134D4 08FF                    <1>       or     bh, bh
3419 000134D6 0F85BA000000            <1>       jnz    sysdma_1
3420                                    <1>
3421 000134DC 20DB                    <1>       and    bl, bl
3422 000134DE 7416                    <1>       jz     short sysdma_01
3423                                    <1>
3424                                    <1>       ; redirect system call to 'sysalloc'
3425 000134E0 89D3                    <1>       mov    ebx, edx ; virtual address of DMA buffer
3426                                    <1>       ;ecx = Buffer size in bytes
3427                                    <1>       ; DMA buffer address <= 16MB upper limit
3428 000134E2 BA00000001              <1>       mov    edx, 1024*1024*16 ; 16MB limit for DMA buff
3429                                    <1>
3430 000134E7 C705[54980100]FFFF-     <1>       mov    dword [dma_addr], 0FFFFFFFFh ; -1
3430 000134EF FFFF                    <1>
3431                                    <1>
3432 000134F1 E9A3E5FFFF              <1>       jmp    sysalloc
3433                                    <1>
3434                                    <1> sysdma_01:
3435 000134F6 B8[00000200]            <1>       mov    eax, sb16_dma_buffer
3436                                    <1>
3437 000134FB 803D[11940100]01        <1>       cmp    byte [audio_device], 1
3438 00013502 722A                    <1>       jb     short sysdma_03
3439                                    <1>
3440 00013504 3B05[30940100]          <1>       cmp    eax, [audio_dma_buff]
3441 0001350A 7507                    <1>       jne    short sysdma_02
3442                                    <1>
3443                                    <1> sysdma_0_err:
3444 0001350C B80B000000              <1>       mov    eax, ERR_PERM_DENIED
3445 00013511 EBB2                    <1>       jmp    short sysdma_perm_err
3446                                    <1>
3447                                    <1> sysdma_02:
3448                                    <1>       ; Only one user is permitted for audio/dma functions
3449                                    <1>
3450 00013513 833D[30940100]00        <1>       cmp    dword [audio_dma_buff], 0
3451 0001351A 7612                    <1>       jna    short sysdma_03
3452                                    <1>
3453 0001351C 8A1D[39940100]          <1>       mov    bl, [audio_user]
3454 00013522 08DB                    <1>       or     bl, bl
3455 00013524 7408                    <1>       jz     short sysdma_03
3456                                    <1>
3457 00013526 3A1D[B3030300]          <1>       cmp    bl, [u.uno]
3458 0001352C 75DE                    <1>       jne    short sysdma_0_err
3459                                    <1>
3460                                    <1> sysdma_03:
3461 0001352E 8A1D[51980100]          <1>       mov    bl, [dma_user]
3462 00013534 20DB                    <1>       and    bl, bl
3463 00013536 750E                    <1>       jnz    short sysdma_04
3464                                    <1>
3465 00013538 8A1D[B3030300]          <1>       mov    bl, [u.uno]
3466 0001353E 881D[51980100]          <1>       mov    [dma_user], bl
3467                                    <1>
3468 00013544 EB15                    <1>       jmp    short sysdma_05
3469                                    <1>
3470                                    <1> sysdma_04:
3471 00013546 8B35[54980100]          <1>       mov    esi, [dma_addr]
3472 0001354C 21F6                    <1>       and    esi, esi
3473 0001354E 740B                    <1>       jz     short sysdma_05
3474                                    <1>
3475 00013550 46                      <1>       inc    esi ; -1 -> 0
3476 00013551 7408                    <1>       jz     short sysdma_05
3477                                    <1>
3478 00013553 3A1D[B3030300]          <1>       cmp    bl, [u.uno]
3479 00013559 75B1                    <1>       jne    short sysdma_0_err
3480                                    <1>
3481                                    <1> sysdma_05:
3482                                    <1>       ; edx = virtual address (user's buffer address)
3483                                    <1>       ;
3484 0001355B 81F900000100            <1>       cmp    ecx, 65536   ; byte count (buffer size)
3485 00013561 0F875BFFFFFF            <1>       ja     sysdma_err
3486                                    <1>       ;
3487 00013567 81C1FF0F0000            <1>       add    ecx, PAGE_SIZE-1 ; 4095
3488 0001356D 6681E100F0              <1>       and    cx, ~PAGE_OFF ; not 4095
3489                                    <1>       ;cmp   ecx, 65536
3490                                    <1>       ;ja    sysdma_err ;
3491 00013572 51                      <1>       push   ecx    ; buffer size (allocated pages * 4096)
3492 00013573 50                      <1>       push   eax    ; offset sb16_dma_buffer
```

```
3493 00013574 89D3              <1>       mov    ebx, edx
3494 00013576 C1E90C            <1>       shr    ecx, 12 ; byte count to page count
3495                            <1>       ; eax = physical address of (audio) dma buffer
3496                            <1>       ; ebx = virtual address of (audio) dma buffer (user's pgdir)
3497                            <1>       ; ecx = page count (>0)
3498 00013579 E87231FFFF        <1>       call   direct_memory_access
3499 0001357E 58                <1>       pop    eax
3500 0001357F 59                <1>       pop    ecx
3501 00013580 0F823CFFFFFF      <1>       jc     sysdma_err
3502                            <1>
3503 00013586 A3[54980100]      <1>       mov    [dma_addr], eax
3504 0001358B 890D[58980100]    <1>       mov    [dma_size], ecx ; dma buffer size (in bytes)
3505                            <1>
3506                            <1>       ;mov   [u.r0], eax ; DMA Buffer Address (Physical)
3507                            <1>
3508                            <1>       ;mov   ebp, [u.usp]  ; ebp points to user's registers
3509                            <1>       ;mov   [ebp+24], ecx ; return to user with ecx value
3510                            <1>
3511                            <1>       ;jmp   sysret
3512                            <1>
3513                            <1>       ; 28/08/2017
3514 00013591 E9C4000000        <1>       jmp    sysdma_51
3515                            <1>
3516                            <1> sysdma_1:
3517 00013596 80FF01            <1>       cmp    bh, 1
3518 00013599 0F87A6000000      <1>       ja     sysdma_5
3519                            <1>
3520 0001359F F6C340            <1>       test   bl, 40h      ; record (read) mode -BL, bit 6-
3521 000135A2 0F851AFFFFFF      <1>       jnz    sysdma_err ; not ready yet!
3522                            <1>
3523 000135A8 A1[54980100]      <1>       mov    eax, [dma_addr] ; physical address of dma buffer
3524 000135AD 21C0              <1>       and    eax, eax
3525 000135AF 0F840DFFFFFF      <1>       jz     sysdma_err
3526                            <1>
3527 000135B5 09D2              <1>       or     edx, edx
3528 000135B7 7504              <1>       jnz    short sysdma_11
3529                            <1>
3530 000135B9 89C2              <1>       mov    edx, eax
3531 000135BB EB08              <1>       jmp    short sysdma_12
3532                            <1> sysdma_11:
3533 000135BD 39C2              <1>       cmp    edx, eax
3534 000135BF 0F82FDFEFFFF      <1>       jb     sysdma_err
3535                            <1> sysdma_12:
3536 000135C5 21C9              <1>       and    ecx, ecx
3537 000135C7 7508              <1>       jnz    short sysdma_13
3538                            <1>
3539 000135C9 8B0D[58980100]    <1>       mov    ecx, [dma_size]
3540 000135CF EB0C              <1>       jmp    short sysdma_14
3541                            <1> sysdma_13:
3542 000135D1 3B0D[58980100]    <1>       cmp    ecx, [dma_size]
3543 000135D7 0F87E5FEFFFF      <1>       ja     sysdma_err
3544                            <1> sysdma_14:
3545 000135DD 89C6              <1>       mov    esi, eax
3546 000135DF 0335[58980100]    <1>       add    esi, [dma_size]
3547                            <1>
3548 000135E5 89D0              <1>       mov    eax, edx
3549 000135E7 01C8              <1>       add    eax, ecx
3550 000135E9 0F82D3FEFFFF      <1>       jc     sysdma_err ; 02/09/2017
3551                            <1>
3552 000135EF 39F0              <1>       cmp    eax, esi
3553 000135F1 0F87CBFEFFFF      <1>       ja     sysdma_err
3554                            <1>
3555 000135F7 8B3D[30940100]    <1>       mov    edi, [audio_dma_buff]
3556 000135FD 8B35[54980100]    <1>       mov    esi, [dma_addr]
3557                            <1>
3558 00013603 09FF              <1>       or     edi, edi
3559 00013605 7424              <1>       jz     short sysdma_16
3560                            <1>
3561 00013607 803D[11940100]01  <1>       cmp    byte [audio_device], 1
3562 0001360E 7208              <1>       jb     short sysdma_15
3563                            <1>
3564                            <1>       ; Sound Blaster 16
3565 00013610 39FE              <1>       cmp    esi, edi
3566 00013612 0F84F4FEFFFF      <1>       je     sysdma_0_err ; permmission denied !
3567                            <1>
3568                            <1> sysdma_15:
3569 00013618 C605[53980100]48  <1>       mov    byte [dma_mode], 48h ; single mode playback
3570                            <1>
3571 0001361F F6C380            <1>       test   bl, 80h ; DMA mode - BL, bit 7, auto init -
3572 00013622 7407              <1>       jz     short sysdma_16
3573                            <1>       ; Auto initialized playback (write) mode
3574 00013624 8005[53980100]10  <1>       add    byte [dma_mode], 10h ; = 58h
3575                            <1> sysdma_16:
3576 0001362B 80E307            <1>       and    bl, 07h
3577 0001362E 881D[52980100]    <1>       mov    [dma_channel], bl
3578 00013634 8915[5C980100]    <1>       mov    [dma_start], edx
3579 0001363A 890D[60980100]    <1>       mov    [dma_count], ecx
3580                            <1>
3581                            <1>       ; 28/08/2017
3582                            <1>       ;call  dma_init
3583                            <1>       ;jmp   sysret
3584 00013640 E94B010000        <1>       jmp    dma_init
3585                            <1>
3586                            <1> sysdma_5:
3587 00013645 80FF05            <1>       cmp    bh, 5
3588 00013648 7223              <1>       jb     short sysdma_3
3589 0001364A 0F87CE000000      <1>       ja     sysdma_6
3590                            <1>
3591                            <1>       ; Get the system's default dma buffer addr and size
3592 00013650 B8[00000200]      <1>       mov    eax, sb16_dma_buffer
3593 00013655 B900000100        <1>       mov    ecx, 65536 ; Buffer size in bytes
3594                            <1>
3595                            <1> sysdma_51:
3596                            <1>       ; 0 = there is not a dma buffer (in use or available)
3597 0001365A A3[64030300]      <1>       mov    [u.r0], eax
```

```
3598                                    <1>
3599 0001365F 8B2D[60030300]            <1>       mov    ebp, [u.usp]  ; ebp points to user's registers
3600 00013665 894D18                    <1>       mov    [ebp+24], ecx ; return to user with ecx value
3601                                    <1>
3602 00013668 E97FA2FFFF                <1>       jmp    sysret
3603                                    <1>
3604                                    <1> sysdma_3:
3605 0001366D 80FF03                    <1>       cmp    bh, 3
3606 00013670 7231                      <1>       jb     short sysdma_2
3607 00013672 776B                      <1>       ja     short sysdma_4
3608                                    <1>
3609                                    <1>       ; Get current dma count down value (remain bytes)
3610                                    <1>       ; 28/08/2017
3611 00013674 0FB635[52980100]          <1>       movzx  esi, byte [dma_channel]
3612 0001367B 0FB696[90410100]          <1>       movzx  edx, byte [dma_flip+esi]
3613 00013682 EE                        <1>       out    dx, al              ; flip-flop clear
3614 00013683 8A96[70410100]            <1>       mov    dl, [dma_cnt+esi] ; dma count register addr
3615 00013689 EC                        <1>       in     al, dx
3616 0001368A 0FB6D8                    <1>       movzx  ebx, al
3617 0001368D EC                        <1>       in     al, dx
3618 0001368E 88C7                      <1>       mov     bh, al
3619                                    <1>
3620 00013690 6683FE04                  <1>       cmp    si, 4 ; channel number ?
3621 00013694 7202                      <1>       jb     short sysdma_31 ; 8 bit dma channel
3622                                    <1>
3623 00013696 D1E3                      <1>       shl    ebx, 1 ; word count to byte count
3624                                    <1>
3625                                    <1> sysdma_31:
3626 00013698 891D[64030300]            <1>       mov    [u.r0], ebx
3627                                    <1>
3628 0001369E E949A2FFFF                <1>       jmp    sysret
3629                                    <1>
3630                                    <1> sysdma_2:
3631                                    <1>       ; Get current dma buffer offset (& page)
3632                                    <1>       ; 28/08/2017
3633 000136A3 0FB635[52980100]          <1>       movzx  esi, byte [dma_channel]
3634 000136AA 0FB696[90410100]          <1>       movzx  edx, byte [dma_flip+esi]
3635 000136B1 EE                        <1>       out    dx, al              ; flip-flop clear
3636 000136B2 8A96[68410100]            <1>       mov    dl, [dma_adr+esi]
3637 000136B8 EC                        <1>       in     al, dx              ; get dma position
3638 000136B9 0FB6D8                    <1>       movzx  ebx, al
3639 000136BC EC                        <1>       in     al, dx
3640 000136BD 88C7                      <1>       mov    bh, al
3641                                    <1>
3642 000136BF 6683FE04                  <1>       cmp    si, 4 ; channel number ?
3643 000136C3 7202                      <1>       jb     short sysdma_21 ; 8 bit dma channel
3644                                    <1>
3645 000136C5 D1E3                      <1>       shl    ebx, 1 ; word offset to byte offset
3646                                    <1>
3647                                    <1> sysdma_21:
3648 000136C7 891D[64030300]            <1>       mov    [u.r0], ebx
3649                                    <1>
3650 000136CD 8A96[78410100]            <1>       mov    dl, [dma_page+esi]
3651 000136D3 EC                        <1>       in     al, dx              ; get dma page
3652                                    <1>
3653                                    <1>       ;add   [u.ro+2], al
3654 000136D4 0805[66030300]            <1>       or     [u.r0+2], al
3655                                    <1>
3656 000136DA E90DA2FFFF                <1>       jmp    sysret
3657                                    <1>
3658                                    <1> sysdma_4:
3659                                    <1>       ; Get current DMA channel number
3660                                    <1>       ; 28/08/2017
3661 000136DF 8A25[51980100]            <1>       mov    ah, [dma_user]
3662 000136E5 20E4                      <1>       and    ah, ah
3663 000136E7 750F                      <1>       jnz    short sysdma_42
3664                                    <1>
3665                                    <1> sysdma_41:
3666                                    <1>       ; Not a valid dma channel (in use)
3667 000136E9 C705[64030300]FFFF-       <1>       mov    dword [u.r0], -1 ; 0FFFFFFFFh
3667 000136F1 FFFF                      <1>
3668 000136F3 E9F4A1FFFF                <1>       jmp    sysret
3669                                    <1>
3670                                    <1> sysdma_42:
3671 000136F8 8B35[54980100]            <1>       mov    esi, [dma_addr]
3672 000136FE 21F6                      <1>       and    esi, esi
3673 00013700 74E7                      <1>       jz     short sysdma_41
3674                                    <1>
3675 00013702 46                        <1>       inc    esi ; -1 -> 0
3676 00013703 74E4                      <1>       jz     short sysdma_41
3677                                    <1>
3678 00013705 A0[52980100]              <1>       mov    al, [dma_channel]
3679                                    <1>
3680 0001370A 3A25[B3030300]            <1>       cmp    ah, [u.uno]
3681 00013710 7502                      <1>       jne    short sysdma_43
3682                                    <1>
3683 00013712 30E4                      <1>       xor    ah, ah ; DMA channel in use by current user
3684                                    <1>
3685                                    <1> sysdma_43:
3686 00013714 A3[64030300]              <1>       mov    [u.r0], eax ; AL = dma channel number
3687                                    <1>                          ; AH > 0 if the the channel
3688                                    <1>                          ; in use by another user/process
3689 00013719 E9CEA1FFFF                <1>       jmp    sysret
3690                                    <1>
3691                                    <1> sysdma_6:
3692 0001371E 80FF06                    <1>       cmp    bh, 6
3693 00013721 7710                      <1>       ja     short sysdma_7
3694                                    <1>
3695                                    <1>       ; 28/08/2017
3696                                    <1>       ; Get current DMA buffer addr and size
3697 00013723 A1[54980100]              <1>       mov    eax, [dma_addr] ; dma buffer address
3698 00013728 8B0D[58980100]            <1>       mov    ecx, [dma_size] ; dma buffer size (in bytes)
3699                                    <1>
3700 0001372E E927FFFFFF                <1>       jmp    sysdma_51
3701                                    <1>
```

```
3702                                    <1> sysdma_7:
3703                                    <1>       ; DMA service STOP
3704 00013733 A0[B3030300]              <1>       mov    al, [u.uno]
3705 00013738 3A05[51980100]            <1>       cmp    al, [dma_user]
3706 0001373E 751D                      <1>       jne    short sysdma_72
3707                                    <1>
3708 00013740 28C0                      <1>       sub    al, al ; 0
3709                                    <1>
3710 00013742 A2[51980100]              <1>       mov    [dma_user], al ; clear user
3711                                    <1>
3712 00013747 8605[53980100]            <1>       xchg   al, [dma_mode]
3713 0001374D 20C0                      <1>       and    al, al
3714                                    <1>       ;jz    short sysdma_err
3715 0001374F 7527                      <1>       jnz    short sysdma_73
3716                                    <1>
3717                                    <1> sysdma_71:
3718 00013751 31C0                      <1>       xor    eax, eax
3719 00013753 A3[64030300]              <1>       mov    [u.r0], eax; 0
3720 00013758 E98FA1FFFF                <1>       jmp    sysret
3721                                    <1>
3722                                    <1> sysdma_72:
3723                                    <1>       ; 28/08/2017
3724 0001375D 803D[51980100]00          <1>       cmp    byte [dma_user], 0
3725 00013764 76EB                      <1>       jna    short sysdma_71 ; Nothing to do !
3726                                    <1>
3727 00013766 833D[54980100]00          <1>       cmp    dword [dma_addr], 0
3728 0001376D 0F8799FDFFFF              <1>       ja     sysdma_0_err
3729                                    <1>
3730 00013773 A2[51980100]              <1>       mov    [dma_user], al ; reset to current user
3731                                    <1>
3732                                    <1> sysdma_73:
3733                                    <1>       ; 28/08/2017
3734 00013778 0FB635[52980100]          <1>       movzx  esi, byte [dma_channel]
3735 0001377F 0FB696[80410100]          <1>       movzx  edx, byte [dma_mask+esi]
3736 00013786 A0[52980100]              <1>       mov    al, [dma_channel]
3737 0001378B 0C04                      <1>       or     al, 4
3738 0001378D EE                        <1>       out    dx, al
3739                                    <1>
3740 0001378E EBC1                      <1>       jmp    short sysdma_71
3741                                    <1>
3742                                    <1> dma_init:
3743                                    <1>       ; 28/08/2017
3744                                    <1>       ; 20/08/2017
3745                                    <1>       ; DMA initialization
3746                                    <1>       ; 14/08/2017
3747                                    <1>       ; 03/08/2017, 06/08/2017, 08/08/2017
3748                                    <1>       ; 02/07/2017, 13/07/2017, 16/07/2017, 30/07/2017
3749                                    <1>       ; (Derived from 'DMA_INIT' procedure in SB16MOD.ASM)
3750                                    <1>       ; Modified for TRDOS 386 DMA buffer allocation & initialization !
3751                                    <1>
3752 00013790 8B1D[5C980100]            <1>       mov    ebx, [dma_start]
3753 00013796 8B0D[60980100]            <1>       mov    ecx, [dma_count]
3754                                    <1>
3755 0001379C 0FB635[52980100]          <1>       movzx  esi, byte [dma_channel]
3756                                    <1>
3757 000137A3 6683FE04                  <1>       cmp    si, 4
3758 000137A7 7205                      <1>       jb     short gdmi1
3759                                    <1>       ; 08/08/2017
3760 000137A9 66D1E9                    <1>       shr    cx, 1 ; word count
3761 000137AC D1EB                      <1>       shr    ebx, 1 ; convert byte offset to word offset
3762                                    <1> gdmi1:
3763                                    <1>       ;mov   [dma_poff], bx ; 08/08/2017
3764 000137AE 6649                      <1>       dec    cx                  ; dma size = block size - 1
3765                                    <1>
3766 000137B0 0FB696[80410100]          <1>       movzx  edx, byte [dma_mask+esi] ; 30/07/2017
3767 000137B7 A0[52980100]              <1>       mov    al, [dma_channel]
3768 000137BC 0C04                      <1>       or     al, 4
3769 000137BE EE                        <1>       out    dx, al              ; dma channel mask
3770                                    <1>
3771 000137BF 30C0                      <1>       xor    al, al ; 0 ; any value ! 08/08/2017
3772 000137C1 8A96[90410100]            <1>       mov    dl, [dma_flip+esi]
3773 000137C7 EE                        <1>       out    dx, al              ; flip-flop clear
3774                                    <1>
3775 000137C8 8A96[88410100]            <1>       mov    dl, [dma_mod+esi]
3776 000137CE A0[52980100]              <1>       mov    al, [dma_channel]  ; 13/07/2017
3777 000137D3 2403                      <1>       and    al, 3
3778                                    <1>       ; 08/08/2017
3779 000137D5 0A05[53980100]            <1>       or     al, [dma_mode] ; 58h    ; dma mode for SB16
3780 000137DB EE                        <1>       out    dx, al
3781                                    <1>
3782 000137DC 8A96[68410100]            <1>       mov    dl, [dma_adr+esi]
3783 000137E2 88D8                      <1>       mov    al, bl
3784 000137E4 EE                        <1>       out    dx, al              ; offset low
3785                                    <1>
3786 000137E5 88F8                      <1>       mov    al, bh
3787 000137E7 EE                        <1>       out    dx, al              ; offset high
3788                                    <1>
3789 000137E8 8A96[70410100]            <1>       mov    dl, [dma_cnt+esi]
3790 000137EE 88C8                      <1>       mov    al, cl
3791 000137F0 EE                        <1>       out    dx, al              ; size low
3792                                    <1>
3793 000137F1 88E8                      <1>       mov    al, ch
3794 000137F3 EE                        <1>       out    dx, al              ; size high
3795                                    <1>
3796 000137F4 8A96[78410100]            <1>       mov    dl, [dma_page+esi]
3797                                    <1>       ; 14/08/2017
3798 000137FA 6683FE04                  <1>       cmp    si, 4
3799 000137FE 7305                      <1>       jnb    short gdmi2
3800 00013800 C1EB10                    <1>       shr    ebx, 16
3801 00013803 EB06                      <1>       jmp    short gdmi3
3802                                    <1> gdmi2:
3803                                    <1>       ; 09/08/2017
3804 00013805 C1EB0F                    <1>       shr    ebx, 15       ; complete 16 bit shift
3805 00013808 80E3FE                    <1>       and    bl, 0FEh ; clear bit 0 (not necessary)
3806                                    <1> gdmi3:
```

```
3807 0001380B 88D8                <1>       mov    al, bl
3808 0001380D EE                  <1>       out    dx, al              ; page
3809                              <1>
3810 0001380E 8A96[80410100]      <1>       mov    dl, [dma_mask+esi]
3811 00013814 A0[52980100]        <1>       mov    al, [dma_channel]  ; 13/07/2017
3812 00013819 2403                <1>       and    al, 3
3813 0001381B EE                  <1>       out    dx, al              ; dma channel unmask
3814                              <1>
3815                              <1>       ;retn
3816                              <1>       ; 28/08/2017
3817 0001381C E9CBA0FFFF          <1>       jmp    sysret
3818                              <1>
3819                              <1> otty:
3820                              <1> sret:
3821                              <1> ocvt:
3822                              <1> ctty:
3823                              <1> cret:
3824                              <1> ccvt:
3825                              <1> rtty:
3826                              <1> wtty:
3827                              <1> rmem:
3828                              <1> wmem:
3829                              <1> rfd:
3830                              <1> rhd:
3831                              <1> wfd:
3832                              <1> whd:
3833                              <1> rlpt:
3834                              <1> wlpt:
3835                              <1> rcvt:
3836                              <1> xmtt:
3837 00013821 C3                   <1>       retn
3087                                  %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: 24/01/2021
   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 00013822 01                  <1>             db 1 ; A: = 0 & B: = 1
  22                              <1>
  23                              <1> Restore_CDIR:
  24 00013823 FF                  <1>             db 0FFh ; Initial value -> any number except 0
  25                              <1>
  26                              <1> msg_CRLF_temp:
  27 00013824 070D0A00            <1>             db 07h, 0Dh, 0Ah, 0
  28                              <1>
  29                              <1> Magic_Bytes:
  30 00013828 04                  <1>             db 4
  31 00013829 01                  <1>             db 1
  32                              <1> mainprog_Version:
  33 0001382A 07                  <1>             db 7
  34 0001382B 5B5452444F535D204D- <1>             db "[TRDOS] Main Program v2.0.240121"
  34 00013834 61696E2050726F6772- <1>
  34 0001383D 616D2076322E302E32- <1>
  34 00013846 3430313231          <1>
  35 0001384B 0D0A                <1>             db 0Dh, 0Ah
  36 0001384D 286329204572646F67- <1>             db "(c) Erdogan Tan 2005-2021"
  36 00013856 616E2054616E203230- <1>
  36 0001385F 30352D32303231      <1>
  37 00013866 0D0A00              <1>             db 0Dh, 0Ah, 0
  38                              <1>
  39                              <1> MainProgCfgFile: ; 14/04/2016
  40 00013869 4D41494E50524F472E- <1>             db "MAINPROG.CFG", 0
  40 00013872 43464700            <1>
  41                              <1>
  42                              <1> TRDOSPromptLabel:
  43 00013876 5452444F53          <1>             db "TRDOS"
  44 0001387B 00                  <1>             db 0
  45 0001387C 00<rept>            <1>                times 5 db 0
  46 00013881 00                  <1>             db 0
  47                              <1>
  48                              <1> ; INTERNAL COMMANDS
  49                              <1> Command_List:
  50 00013882 44495200            <1> Cmd_Dir:    db "DIR", 0
  51 00013886 434400              <1> Cmd_Cd:           db "CD", 0
  52 00013889 433A00              <1> Cmd_Drive:  db "C:", 0
  53 0001388C 56455200            <1> Cmd_Ver:    db "VER", 0
  54 00013890 4558495400          <1> Cmd_Exit:   db "EXIT", 0
  55 00013895 50524F4D505400      <1> Cmd_Prompt: db "PROMPT", 0
  56 0001389C 564F4C554D4500      <1> Cmd_Volume: db "VOLUME", 0
  57 000138A3 4C4F4E474E414D4500  <1> Cmd_LongName:     db "LONGNAME", 0
  58 000138AC 4441544500          <1> Cmd_Date:   db "DATE", 0
  59 000138B1 54494D4500          <1> Cmd_Time:   db "TIME", 0
  60 000138B6 52554E00            <1> Cmd_Run:    db "RUN", 0
  61 000138BA 53455400            <1> Cmd_Set:    db "SET", 0
  62 000138BE 434C5300            <1> Cmd_Cls:    db "CLS", 0
  63 000138C2 53484F5700          <1> Cmd_Show:   db "SHOW", 0
  64 000138C7 44454C00            <1> Cmd_Del:    db "DEL", 0
  65 000138CB 41545452494200      <1> Cmd_Attrib: db "ATTRIB", 0
  66 000138D2 52454E414D4500      <1> Cmd_Rename: db "RENAME", 0
  67 000138D9 524D44495200        <1> Cmd_Rmdir:  db "RMDIR", 0
```

```
00013DF 4D4B44495200        <1> Cmd_Mkdir:   db "MKDIR", 0
000138E5 434F505900          <1> Cmd_Copy:    db "COPY", 0
000138EA 4D4F564500          <1> Cmd_Move:    db "MOVE", 0
000138EF 5041544800          <1> Cmd_Path:    db "PATH", 0
000138F4 4D454D00            <1> Cmd_Mem:     db "MEM", 0
000138F8 00                  <1>              db 0
000138F9 46494E4400          <1> Cmd_Find:    db "FIND", 0
000138FE 4543484F00          <1> Cmd_Echo:    db "ECHO", 0
00013903 2A00                <1> Cmd_Remark:  db "*", 0
00013905 3F00                <1> Cmd_Help:    db "?", 0
00013907 44455649434500      <1> Cmd_Device:  db "DEVICE", 0
0001390E 4445564C49535400    <1> Cmd_DevList: db "DEVLIST", 0
00013916 434844495200        <1> Cmd_Chdir:   db "CHDIR", 0
0001391C 4245455000          <1> Cmd_Beep:    db "BEEP", 0
                             <1>
00013921 00                  <1>              db 0
                             <1>
                             <1> ; 15/02/2016 (FILE.ASM, 09/10/2011)
                             <1> invalid_fname_chars:
00013922 222728292A2B2C2F    <1>              db 22h, 27h, 28h, 29h, 2Ah, 2Bh, 2Ch, 2Fh
0001392A 3A3B3C3D3E3F40      <1>              db 3Ah, 3Bh, 3Ch, 3Dh, 3Eh, 3Fh, 40h
00013931 5B5C5D5E60          <1>              db 5Bh, 5Ch, 5Dh, 5Eh, 60h
                             <1> sizeInvFnChars  equ ($ - invalid_fname_chars)
                             <1> ;
                             <1>
                             <1> Msg_Enter_Date:
00013936 456E746572206E6577- <1>                 db 'Enter new date (dd-mm-yy): '
0001393F 206461746520286464- <1>
00013948 2D6D6D2D7979293A20  <1>
00013951 00                  <1>                 db 0
                             <1> Msg_Show_Date:
00013952 43757272656E742064- <1>                 db   'Current date is '
0001395B 61746520697320      <1>
00013962 30                  <1> Day:            db   '0'
00013963 30                  <1>              db   '0'
00013964 2F                  <1>                 db   '/'
00013965 30                  <1> Month:          db   '0'
00013966 30                  <1>              db   '0'
00013967 2F                  <1>                 db   '/'
00013968 30                  <1> Century:        db   '0'
00013969 30                  <1>                 db   '0'
0001396A 30                  <1> Year:           db   '0'
0001396B 30                  <1>              db   '0'
0001396C 0D0A00              <1>                 db   0Dh, 0Ah, 0
                             <1>
                             <1> Msg_Enter_Time:
0001396F 456E746572206E6577- <1>              db 'Enter new time: '
00013978 2074696D653A20      <1>
0001397F 00                  <1>              db 0
                             <1> Msg_Show_Time:
00013980 43757272656E742074- <1>              db   'Current time is '
00013989 696D6520697320      <1>
00013990 30                  <1> Hour:           db   '0'
00013991 30                  <1>              db   '0'
00013992 3A                  <1>              db   ':'
00013993 30                  <1> Minute:         db   '0'
00013994 30                  <1>              db   '0'
00013995 3A                  <1>              db   ':'
00013996 30                  <1> Second:         db   '0'
00013997 30                  <1>              db   '0'
00013998 0D0A00              <1>              db   0Dh, 0Ah, 0
                             <1>
                             <1> ;VolSize_Unit1:   dd 0
                             <1> ;VolSize_Unit2:   dd 0
                             <1>
                             <1> VolSize_KiloBytes:
0001399B 206B696C6F62797465- <1>              db " kilobytes", 0Dh, 0Ah, 0
000139A4 730D0A00            <1>
                             <1> VolSize_Bytes:
000139A8 2062797465730D0A00  <1>              db " bytes", 0Dh, 0Ah, 0
                             <1> Volume_in_drive:
000139B1 0D0A                <1>              db 0Dh, 0Ah
                             <1> Vol_FS_Name:
000139B3 54522046533120      <1>              db "TR FS1 "
000139BA 566F6C756D6520696E- <1>              db "Volume in drive "
000139C3 20647269766520      <1>
000139CA 30                  <1> Vol_Drv_Name:   db 30h
000139CB 3A                  <1>              db ":"
000139CC 20697320            <1>              db " is "
000139D0 0D0A00              <1>              db 0Dh, 0Ah, 0
                             <1> Dir_Drive_Str:
000139D3 54522D444F53204472- <1>                  db "TR-DOS Drive "
000139DC 69766520            <1>
                             <1> Dir_Drive_Name:
000139E0 303A                <1>                  db "0:"
000139E2 0D0A                <1>                  db  0Dh, 0Ah
                             <1> Vol_Str_Header:
000139E4 566F6C756D65204E61- <1>                  db "Volume Name: "
000139ED 6D653A20            <1>
                             <1> Vol_Name:
000139F1 00<rept>            <1>              times 64 db 0
00013A31 00                  <1>              db 0
                             <1> Vol_Serial_Header:
00013A32 0D0A                <1>              db 0Dh, 0Ah
00013A34 566F6C756D65205365- <1>              db "Volume Serial No: "
00013A3D 7269616C204E6F3A20  <1>
                             <1> Vol_Serial1:
00013A46 30303030            <1>              db "0000"
00013A4A 2D                  <1>              db "-"
                             <1> Vol_Serial2:
00013A4B 30303030            <1>              db "0000"
00013A4F 0D0A00              <1>              db 0Dh, 0Ah, 0
                             <1>
                             <1> ;Vol_Tot_Sec_Str_Start:
                             <1> ;            dd 0
```

```
                            <1> Vol_Total_Sector_Header:
00013A52 0D0A               <1>             db 0Dh, 0Ah
00013A54 566F6C756D65205369- <1>             db "Volume Size : ", 0
00013A5D 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:
00013A63 467265652053706163- <1>             db "Free Space  : ", 0
00013A6C 6520203A2000        <1>
                            <1> ;Vol_Free_Sectors_Str:
                            <1> ;           db "0000000000"
                            <1> ;Vol_Free_Sectors_Str_End:
                            <1> ;           db 0
                            <1>
                            <1> Dir_Str_Header:
00013A72 4469726563746F7279- <1>                 db "Directory: "
00013A7B 3A20                <1>
00013A7D 2F                 <1> Dir_Str_Root:   db "/"
00013A7E 00<rept>           <1> Dir_Str:        times 64 db 0
00013ABE 00000000           <1>                 dd 0
00013AC2 00                 <1>                 db 0
                            <1>
                            <1> Msg_Bad_Command:
00013AC3 42616420636F6D6D61- <1>                 db "Bad command or file name!"
00013ACC 6E64206F722066696C- <1>
00013AD5 65206E616D6521      <1>
00013ADC 0D0A00             <1>                 db 0Dh, 0Ah, 0
                            <1>
                            <1> msgl_drv_not_ready:
00013ADF 070D0A             <1>             db 07h, 0Dh, 0Ah
                            <1>
                            <1> ; CMD_INTR.ASM - 09/11/2011 - Messages
                            <1>
                            <1> Msg_Not_Ready_Read_Err:
00013AE2 4472697665206E6F74- <1>                 db "Drive not ready or read error!"
00013AEB 207265616479206F72- <1>
00013AF4 207265616420657272- <1>
00013AFD 6F7221              <1>
00013B00 0D0A00             <1>                 db 0Dh, 0Ah, 0
                            <1>
                            <1> Msg_Not_Ready_Write_Err:
00013B03 4472697665206E6F74- <1>                 db "Drive not ready or write error!"
00013B0C 207265616479206F72- <1>
00013B15 207772697465206572- <1>
00013B1E 726F7221            <1>
00013B22 0D0A00             <1>                 db 0Dh, 0Ah, 0
                            <1>
                            <1> Msg_Dir_Not_Found:
00013B25 4469726563746F7279- <1>                 db "Directory not found!"
00013B2E 206E6F7420666F756E- <1>
00013B37 6421                <1>
00013B39 0D0A00             <1>                 db 0Dh, 0Ah, 0
                            <1>
                            <1> Msg_File_Not_Found:
00013B3C 46696C65206E6F7420- <1>                 db "File not found!"
00013B45 666F756E6421        <1>
00013B4B 0D0A00             <1>                 db 0Dh, 0Ah, 0
                            <1>
                            <1> Msg_File_Directory_Not_Found:
00013B4E 46696C65206F722064- <1>                 db "File or directory not found!"
00013B57 69726563746F727920- <1>
00013B60 6E6F7420666F756E64- <1>
00013B69 21                  <1>
00013B6A 0D0A00             <1>                 db 0Dh, 0Ah, 0
                            <1>
                            <1> Msg_LongName_Not_Found:
00013B6D 4C6F6E67206E616D65- <1>                 db "Long name not found!"
00013B76 206E6F7420666F756E- <1>
00013B7F 6421                <1>
00013B81 0D0A00             <1>                 db 0Dh, 0Ah, 0
                            <1>
                            <1> beep_Insufficient_Memory: ; 20/02/2017
00013B84 0D0A               <1>             db 0Dh, 0Ah
00013B86 07                 <1>             db 07h
                            <1> Msg_Insufficient_Memory:
00013B87 496E73756666696369- <1>                 db "Insufficient memory!"
00013B90 656E74206D656D6F72- <1>
00013B99 7921                <1>
00013B9B 0D0A00             <1>                 db 0Dh, 0Ah, 0
                            <1>
                            <1> Msg_Error_Code:
00013B9E 436F6D6D616E642066- <1>                 db 'Command failed! Error code : '
00013BA7 61696C656421204572- <1>
00013BB0 726F7220636F646520- <1>
00013BB9 3A20                <1>
00013BBB 303068             <1> error_code_hex: db '00h'
00013BBE 0A0A00             <1>                 db 0Ah, 0Ah, 0
                            <1>
00013BC1 90                 <1> align 2
                            <1>
                            <1> ; 10/02/2016
                            <1> ; DIR.ASM - 09/10/2011
                            <1>
00013BC2 3C4449523E20202020- <1> Type_Dir:       db '<DIR>     ' ; 10 bytes
00013BCB 20                  <1>
                            <1>
                            <1> File_Name:
00013BCC 20<rept>           <1>                 times 12 db 20h
00013BD8 20                 <1>             db 20h
                            <1> Dir_Or_FileSize:
00013BD9 20<rept>           <1>                 times 10 db 20h
```

```
00013BE3 20                  <1>             db 20h
                             <1> File_Attribute:
00013BE4 20202020            <1>             dd 20202020h
00013BE8 20                  <1>             db 20h
                             <1> File_Day:
00013BE9 3030                <1>                 db '0','0'
00013BEB 2F                  <1>             db '/'
                             <1> File_Month:
00013BEC 3030                <1>                 db '0','0'
00013BEE 2F                  <1>             db '/'
                             <1> File_Year:
00013BEF 30303030            <1>                 db '0','0','0','0'
00013BF3 20                  <1>             db 20h
                             <1> File_Hour:
00013BF4 3030                <1>                 db '0','0'
00013BF6 3A                  <1>             db ':'
                             <1> File_Minute:
00013BF7 3030                <1>                 db '0','0'
00013BF9 00                  <1>             db 0
                             <1>
                             <1> Decimal_File_Count_Header:
00013BFA 0D0A                <1>             db 0Dh, 0Ah
                             <1> Decimal_File_Count:
00013BFC 00<rept>            <1>             times 6 db 0
                             <1>
00013C02 2066696C6528732920- <1> str_files:  db " file(s) & "
00013C0B 2620                <1>
                             <1> Decimal_Dir_Count:
00013C0D 00<rept>            <1>             times 6 db 0
                             <1> str_dirs:
00013C13 206469726563746F72- <1>             db " directory(s) "
00013C1C 7928732920          <1>
00013C21 0D0A00              <1>             db 0Dh, 0Ah, 0
                             <1>
00013C24 206279746528732920- <1> str_bytes:  db " byte(s) in file(s)"
00013C2D 696E2066696C652873- <1>
00013C36 29                  <1>
00013C37 0D0A00              <1>             db 0Dh, 0Ah, 0
                             <1>
                             <1> ; CMD_INTR.ASM - 09/11/2011
                             <1> ; 07/10/2010
                             <1> Msg_invalid_name_chars:
00013C3A 496E76616C69642066- <1>                 db "Invalid file or directory name characters!"
00013C43 696C65206F72206469- <1>
00013C4C 726563746F7279206E- <1>
00013C55 616D65206368617261- <1>
00013C5E 637465727321        <1>
00013C64 0D0A00              <1>             db 0Dh, 0Ah, 0
                             <1> ; 21/02/2016
00013C67 46696C65206F722064- <1> Msg_Name_Exists: db "File or directory name exists!"
00013C70 69726563746F727920- <1>
00013C79 6E616D652065786973- <1>
00013C82 747321              <1>
00013C85 0D0A00              <1>                 db 0Dh, 0Ah, 0
                             <1> Msg_DoYouWantMkdir:
00013C88 446F20796F75207761- <1>                 db "Do you want to make directory ", 0
00013C91 6E7420746F206D616B- <1>
00013C9A 65206469726563746F- <1>
00013CA3 72792000            <1>
00013CA7 2028592F4E29203F20- <1> Msg_YesNo:      db " (Y/N) ? ", 0
00013CB0 00                  <1>
00013CB1 000D0A00            <1> Y_N_nextline:      db 0, 0Dh, 0Ah, 0
00013CB5 4F4B2E0D0A00        <1> Msg_OK:            db "OK.", 0Dh, 0Ah, 0
                             <1>
                             <1> ; 27/02/2016
                             <1> Msg_DoYouWantRmDir:
00013CBB 446F20796F75207761- <1>                 db "Do you want to delete directory ", 0
00013CC4 6E7420746F2064656C- <1>
00013CCD 657465206469726563- <1>
00013CD6 746F72792000        <1>
                             <1> Msg_Dir_Not_Empty:
00013CDC 4469726563746F7279- <1>                 db "Directory not empty!"
00013CE5 206E6F7420656D7074- <1>
00013CEE 7921                <1>
00013CF0 0D0A00              <1>                 db 0Dh, 0Ah, 0
                             <1>
                             <1> Msg_DoYouWantDelete:
00013CF3 446F20796F75207761- <1>                 db "Do you want to delete file ",0
00013CFC 6E7420746F2064656C- <1>
00013D05 6574652066696C6520- <1>
00013D0E 00                  <1>
                             <1>
00013D0F 44656C657465642E2E- <1> Msg_Deleted:    db "Deleted...", 0Dh, 0Ah, 0
00013D18 2E0D0A00            <1>
                             <1>
                             <1> Msg_Permission_Denied:
00013D1C 07                  <1>                 db 7
00013D1D 5065726D697373696F- <1>                 db "Permission denied!", 0Dh, 0Ah, 0
00013D26 6E2064656E69656421- <1>
00013D2F 0D0A00              <1>
                             <1>
                             <1> ; 04/03/2016
00013D32 4E657720            <1> Msg_New:        db "New "
00013D36 00                  <1>                 db 0
                             <1> Str_Attributes:
00013D37 417474726962757465- <1>                 db "Attributes : "
00013D40 73203A20            <1>
00013D44 4E4F524D414C        <1> Attr_Chars:     db "NORMAL"
00013D4A 00                  <1>                 db 0
                             <1>
                             <1> ; 06/03/2016
                             <1> ; CMD_INTR.ASM - 16/11/2010
                             <1> Msg_DoYouWantRename:
00013D4B 446F20796F75207761- <1>                 db "Do you want to rename ", 0
00013D54 6E7420746F2072656E- <1>
```

```
00013D5D 616D652000          <1>
00013D62 66696C652000        <1> Rename_File:    db "file ", 0
00013D68 6469726563746F7279- <1> Rename_Directory: db "directory ", 0
00013D71 2000                <1>
00013D73 00<rept>            <1> Rename_OldName: times 13 db 0
00013D80 20617320            <1> Msg_File_rename_as: db " as "
00013D84 00<rept>            <1> Rename_NewName: times 13 db 0
                             <1>
                             <1> ; 08/03/2016
                             <1> ; CMD_INTR.ASM - 01/08/2010 - 23/04/2011
                             <1> msg_not_same_drv:
00013D91 4E6F742073616D6520- <1>                db "Not same drive!"
00013D9A 647269766521        <1>
00013DA0 0D0A00              <1>                db 0Dh, 0Ah, 0
                             <1>
                             <1> Msg_DoYouWantMoveFile:
00013DA3 446F20796F75207761- <1>                db "Do you want to move file", 0
00013DAC 6E7420746F206D6F76- <1>
00013DB5 652066696C6500      <1>
                             <1>
                             <1> msg_insufficient_disk_space:
00013DBC 496E73756666696369- <1>                db "Insufficient disk space!"
00013DC5 656E74206469736B20- <1>
00013DCE 737061636521        <1>
00013DD4 0D0A00              <1>                db 0Dh, 0Ah, 0
                             <1>
                             <1> ; 01/08/2010
                             <1> msg_source_file:
00013DD7 0D0A536F7572636520- <1>                db 0Dh, 0Ah, "Source file name      :   "
00013DE0 66696C65206E616D65- <1>
00013DE9 2020202020203A2020- <1>
00013DF2 20                  <1>
                             <1> msg_source_file_drv:
00013DF3 203A00              <1>                db " :", 0
                             <1> msg_destination_file:
00013DF6 0D0A44657374696E61- <1>                db 0Dh, 0Ah, "Destination file name :   "
00013DFF 74696F6E2066696C65- <1>
00013E08 206E616D65203A2020- <1>
00013E11 20                  <1>
                             <1> msg_destination_file_drv:
00013E12 203A00              <1>                db " :", 0
                             <1> msg_copy_nextline:
00013E15 0D0A00              <1>                db 0Dh, 0Ah, 0
                             <1>
                             <1> ; 15/03/2016
                             <1> ; CMD_INTR.ASM
                             <1>
                             <1> Msg_DoYouWantOverWriteFile:
00013E18 446F20796F75207761- <1>                    db "Do you want to overwrite file ",0
00013E21 6E7420746F206F7665- <1>
00013E2A 727772697465206669- <1>
00013E33 6C652000            <1>
                             <1>
                             <1> Msg_DoYouWantCopyFile:
00013E37 446F20796F75207761- <1>                    db "Do you want to copy file",0
00013E40 6E7420746F20636F70- <1>
00013E49 792066696C6500      <1>
                             <1>
                             <1> Msg_read_file_error_before_EOF:
00013E50 46696C652072656164- <1>                db "File reading error! (before EOF)"
00013E59 696E67206572726F72- <1>
00013E62 2120286265666F7265- <1>
00013E6B 20454F4629          <1>
00013E70 0A0A00              <1>                db 0Ah, 0Ah, 0
                             <1>
                             <1> ; 18/03/2016
                             <1> ; TRDOS 386 (v2.0) mainprog copy procedure
                             <1> msg_reading:
00013E73 52656164696E672E2E- <1>                db "Reading... ", 0
00013E7C 2E2000              <1>
                             <1> msg_writing:
00013E7F 57726974696E672E2E- <1>                db "Writing... ", 0
00013E88 2E2000              <1>
                             <1> percentagestr:
00013E8B 2020202500          <1>                db "   %", 0  ; "  0%" .. "100%"
                             <1> ; 11/04/2016
                             <1> Msg_No_Set_Space:
00013E90 496E73756666696369- <1>                    db "Insufficient environment space!"
00013E99 656E7420656E766972- <1>
00013EA2 6F6E6D656E74207370- <1>
00013EAB 61636521            <1>
00013EAF 0D0A00              <1>                    db 0Dh, 0Ah, 0
                             <1> ; 18/04/2016
                             <1> isc_msg:
00013EB2 0D0A                <1>                db 0Dh, 0Ah
00013EB4 494E56414C49442053- <1>                db "INVALID SYSTEM CALL", 0
00013EBD 595354454D2043414C- <1>
00013EC6 4C00                <1>
                             <1> usi_msg:
00013EC8 0D0A                <1>                db 0Dh, 0Ah
00013ECA 554E444546494E4544- <1>                db "UNDEFINED SOFTWARE INTERRUPT", 0
00013ED3 20534F465457415245- <1>
00013EDC 20494E544552525550- <1>
00013EE5 5400                <1>
                             <1> ifc_msg:
00013EE7 0D0A                <1>                db 0Dh, 0Ah
00013EE9 494E56414C49442046- <1>                db "INVALID FUNCTION CALL"
00013EF2 554E4354494F4E2043- <1>
00013EFB 414C4C              <1>
                             <1> inv_msg_for_trdos_v2:
00013EFE 20                  <1>                db 20h
00013EFF 666F72205452444F53- <1>                db "for TRDOS v2!"
00013F08 20763221            <1>
00013F0C 07                  <1>                db 07h
00013F0D 0D0A                <1>                db 0Dh, 0Ah
```

```
00013F0F 0D0A                <1>             db 0Dh, 0Ah
00013F11 494E5420            <1>             db "INT "
00013F15 303068              <1> int_num_str: db "00h"
00013F18 0D0A                <1>             db 0Dh, 0Ah
00013F1A 454158203A20        <1>             db "EAX : "
00013F20 3030303030303030-   <1> eax_str:    db "00000000h", 0Dh, 0Ah
00013F29 0D0A                <1>
00013F2B 454950203A20        <1>             db "EIP : "
00013F31 3030303030303030-   <1> eip_str:    db "00000000h", 0Dh, 0Ah, 0
00013F3A 0D0A00              <1>
                             <1>
                             <1> ; 07/10/2016
                             <1> ; Device names & parameters (for kernel devices)
                             <1>
00013F3D 90                  <1> align 2
                             <1> KDEV_NAME:
00013F3E 5454590000000000    <1>             db 'TTY',0,0,0,0,0 ; 1
00013F46 4D454D0000000000    <1>             db 'MEM',0,0,0,0,0 ; 2
00013F4E 4644300000000000    <1>             db 'FD0',0,0,0,0,0 ; 3
00013F56 4644310000000000    <1>             db 'FD1',0,0,0,0,0 ; 4
00013F5E 4844300000000000    <1>             db 'HD0',0,0,0,0,0 ; 5
00013F66 4844310000000000    <1>             db 'HD1',0,0,0,0,0 ; 6
00013F6E 4844320000000000    <1>             db 'HD2',0,0,0,0,0 ; 7
00013F76 4844330000000000    <1>             db 'HD3',0,0,0,0,0 ; 8
00013F7E 4C50540000000000    <1>             db 'LPT',0,0,0,0,0 ; 9
00013F86 5454593000000000    <1>             db 'TTY0',0,0,0,0 ; 10
00013F8E 5454593100000000    <1>             db 'TTY1',0,0,0,0 ; 11
00013F96 5454593200000000    <1>             db 'TTY2',0,0,0,0 ; 12
00013F9E 5454593300000000    <1>             db 'TTY3',0,0,0,0 ; 13
00013FA6 5454593400000000    <1>             db 'TTY4',0,0,0,0 ; 14
00013FAE 5454593500000000    <1>             db 'TTY5',0,0,0,0 ; 15
00013FB6 5454593600000000    <1>             db 'TTY6',0,0,0,0 ; 16
00013FBE 5454593700000000    <1>             db 'TTY7',0,0,0,0 ; 17
00013FC6 5454593800000000    <1>             db 'TTY8',0,0,0,0 ; 18
00013FCE 5454593900000000    <1>             db 'TTY9',0,0,0,0 ; 19
00013FD6 434F4D3100000000    <1>             db 'COM1',0,0,0,0 ; 18
00013FDE 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
                             <1>             ;db 'SD2',0,0,0,0,0 ; 37
                             <1>             ;db 'SD3',0,0,0,0,0 ; 38
                             <1>             ;db 'SATA0'    ; 35
                             <1>             ;db 'SATA1'    ; 36
                             <1>             ;db 'SATA2'        ; 37
                             <1>             ;db 'SATA3'        ; 38
                             <1>             ;db 'PATA0',0,0,0  ; 5
                             <1>             ;db 'PATA1',0,0,0  ; 6
                             <1>             ;db 'PATA2',0,0,0  ; 7
                             <1>             ;db 'PATA3',0,0,0  ; 8
                             <1>             ;db 'WIRELESS'        ; 39
                             <1>             ;db 'HDMI',0,0,0,0 ; 40
00013FE6 4E554C4C00000000    <1>             db 'NULL',0,0,0,0 ; 0
                             <1>
                             <1> NumOfKernelDevNames equ ($-KDEV_NAME) / 8 ; 20 (07/10/2016)
                             <1>
                             <1> KDEV_NUMBER:
00013FEE 010203040506070809  <1>             db 1,2,3,4,5,6,7,8,9
00013FF7 0A0B0C0D0E0F101112- <1>             db 10,11,12,13,14,15,16,17,18,19
00014000 13                  <1>
00014001 121300              <1>             db 18,19,0
                             <1>
                             <1> NumOfKernelDevices equ $ - KDEV_NUMBER
                             <1>
                             <1> KDEV_OADDR:
00014004 [21380100]          <1>             dd otty ;tty  ; 1
00014008 [21380100]          <1>             dd sret ;mem  ; 2
0001400C [21380100]          <1>             dd sret ;fd0  ; 3
00014010 [21380100]          <1>             dd sret ;fd1  ; 4
00014014 [21380100]          <1>             dd sret ;hd0  ; 5
00014018 [21380100]          <1>             dd sret ;hd1  ; 6
0001401C [21380100]          <1>             dd sret ;hd2  ; 7
00014020 [21380100]          <1>             dd sret ;hd3  ; 8
00014024 [21380100]          <1>             dd sret ;lpt  ; 9
00014028 [21380100]          <1>             dd ocvt ;tty0 ; 10
0001402C [21380100]          <1>             dd ocvt ;tty1 ; 11
00014030 [21380100]          <1>             dd ocvt ;tty2 ; 12
00014034 [21380100]          <1>             dd ocvt ;tty3 ; 13
00014038 [21380100]          <1>             dd ocvt ;tty4 ; 14
0001403C [21380100]          <1>             dd ocvt ;tty5 ; 15
00014040 [21380100]          <1>             dd ocvt ;tty6 ; 16
00014044 [21380100]          <1>             dd ocvt ;tty7 ; 17
00014048 [21380100]          <1>             dd ocvt ;tty8 ; 18
0001404C [21380100]          <1>             dd ocvt ;tty9 ; 19
```

```
493                             <1>             ;dd ocvt ;com1 ; 18
494                             <1>             ;dd ocvt ;com2 ; 19
495 00014050 [21380100]         <1>             dd sret ;null ; 20
496                             <1> KDEV_CADDR:
497 00014054 [21380100]         <1>             dd ctty ;tty  ; 1
498 00014058 [21380100]         <1>             dd cret ;mem  ; 2
499 0001405C [21380100]         <1>             dd cret ;fd0  ; 3
500 00014060 [21380100]         <1>             dd cret ;fd1  ; 4
501 00014064 [21380100]         <1>             dd cret ;hd0  ; 5
502 00014068 [21380100]         <1>             dd cret ;hd1  ; 6
503 0001406C [21380100]         <1>             dd cret ;hd2  ; 7
504 00014070 [21380100]         <1>             dd cret ;hd3  ; 8
505 00014074 [21380100]         <1>             dd cret ;lpt  ; 9
506 00014078 [21380100]         <1>             dd ocvt ;tty0 ; 10
507 0001407C [21380100]         <1>             dd ccvt ;tty1 ; 11
508 00014080 [21380100]         <1>             dd ccvt ;tty2 ; 12
509 00014084 [21380100]         <1>             dd ccvt ;tty3 ; 13
510 00014088 [21380100]         <1>             dd ccvt ;tty4 ; 14
511 0001408C [21380100]         <1>             dd ccvt ;tty5 ; 15
512 00014090 [21380100]         <1>             dd ccvt ;tty6 ; 16
513 00014094 [21380100]         <1>             dd ccvt ;tty7 ; 17
514 00014098 [21380100]         <1>             dd ccvt ;tty8 ; 18
515 0001409C [21380100]         <1>             dd ccvt ;tty9 ; 19
516                             <1>             ;dd ccvt ;com1 ; 18
517                             <1>             ;dd ccvt ;com2 ; 19
518 000140A0 [21380100]         <1>             dd cret ;null ; 20
519                             <1>
520                             <1> KDEV_RADDR:
521 000140A4 [21380100]         <1>             dd rtty ;tty  ; 1
522 000140A8 [21380100]         <1>             dd rmem ;mem  ; 2
523 000140AC [21380100]         <1>             dd rfd  ;fd0  ; 3
524 000140B0 [21380100]         <1>             dd rfd  ;fd1  ; 4
525 000140B4 [21380100]         <1>             dd rhd  ;hd0  ; 5
526 000140B8 [21380100]         <1>             dd rhd  ;hd1  ; 6
527 000140BC [21380100]         <1>             dd rhd  ;hd2  ; 7
528 000140C0 [21380100]         <1>             dd rhd  ;hd3  ; 8
529 000140C4 [21380100]         <1>             dd rlpt ;lpt  ; 9
530 000140C8 [21380100]         <1>             dd rcvt ;tty0 ; 10
531 000140CC [21380100]         <1>             dd rcvt ;tty1 ; 11
532 000140D0 [21380100]         <1>             dd rcvt ;tty2 ; 12
533 000140D4 [21380100]         <1>             dd rcvt ;tty3 ; 13
534 000140D8 [21380100]         <1>             dd rcvt ;tty4 ; 14
535 000140DC [21380100]         <1>             dd rcvt ;tty5 ; 15
536 000140E0 [21380100]         <1>             dd rcvt ;tty6 ; 16
537 000140E4 [21380100]         <1>             dd rcvt ;tty7 ; 17
538 000140E8 [21380100]         <1>             dd rcvt ;tty8 ; 18
539 000140EC [21380100]         <1>             dd rcvt ;tty9 ; 19
540                             <1>             ;dd rcvt ;com1 ; 18
541                             <1>             ;dd rcvt ;com2 ; 19
542 000140F0 [052C0100]         <1>             dd rnull ;null ; 20
543                             <1> KDEV_WADDR:
544 000140F4 [21380100]         <1>             dd wtty ;tty  ; 1
545 000140F8 [21380100]         <1>             dd wmem ;mem  ; 2
546 000140FC [21380100]         <1>             dd wfd  ;fd0  ; 3
547 00014100 [21380100]         <1>             dd wfd  ;fd1  ; 4
548 00014104 [21380100]         <1>             dd whd  ;hd0  ; 5
549 00014108 [21380100]         <1>             dd whd  ;hd1  ; 6
550 0001410C [21380100]         <1>             dd whd  ;hd2  ; 7
551 00014110 [21380100]         <1>             dd whd  ;hd3  ; 8
552 00014114 [21380100]         <1>             dd wlpt ;lpt  ; 9
553 00014118 [21380100]         <1>             dd xmtt ;tty0 ; 10
554 0001411C [21380100]         <1>             dd xmtt ;tty1 ; 11
555 00014120 [21380100]         <1>             dd xmtt ;tty2 ; 12
556 00014124 [21380100]         <1>             dd xmtt ;tty3 ; 13
557 00014128 [21380100]         <1>             dd xmtt ;tty4 ; 14
558 0001412C [21380100]         <1>             dd xmtt ;tty5 ; 15
559 00014130 [21380100]         <1>             dd xmtt ;tty6 ; 16
560 00014134 [21380100]         <1>             dd xmtt ;tty7 ; 17
561 00014138 [21380100]         <1>             dd xmtt ;tty8 ; 18
562 0001413C [21380100]         <1>             dd xmtt ;tty9 ; 19
563                             <1>             ;dd xmtt ;com1 ; 18
564                             <1>             ;dd xmtt ;com2 ; 19
565 00014140 [062C0100]         <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 00014144 07                 <1>             db  00000111b; tty, 1
579 00014145 07                 <1>             db  00000111b; mem, 2
580 00014146 8F                 <1>             db  10001111b; fd0, 3
581 00014147 8F                 <1>             db  10001111b; fd1, 4
582 00014148 8F                 <1>             db  10001111b; hd0, 5
583 00014149 8F                 <1>             db  10001111b; hd1, 6
584 0001414A 8F                 <1>             db  10001111b; hd2, 7
585 0001414B 8F                 <1>             db  10001111b; hd3, 8
586 0001414C 07                 <1>             db  00000111b   ; lpt, 9
587 0001414D 07                 <1>             db  00000111b; tty0, 10
588 0001414E 07                 <1>             db  00000111b; tty1, 11
589 0001414F 07                 <1>             db  00000111b; tty2, 12
590 00014150 07                 <1>             db  00000111b; tty3, 13
591 00014151 07                 <1>             db  00000111b; tty4, 14
592 00014152 07                 <1>             db  00000111b; tty5, 15
593 00014153 07                 <1>             db  00000111b; tty6, 16
594 00014154 07                 <1>             db  00000111b; tty7, 17
595 00014155 07                 <1>             db  00000111b; tty8, 18
596 00014156 07                 <1>             db  00000111b; tty9, 19
597                             <1>             ;db 00000111b; com1, 18
```

```
 598                                    <1>             ;db 00000111b; com2, 19
 599 00014157 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 00014158 000000010203000400-       <1>       db  0,0,0,1,2,3,0,4,0,5,6,7,8,9,0,0
 615 00014161 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 00014168 00020406C0C4C8CC          <1> dma_adr:    db 0,2,4,6,0C0h,0C4h,0C8h,0CCh
 622 00014170 01030507C2C6CACE          <1> dma_cnt:    db 1,3,5,7,0C2h,0C6h,0CAh,0CEh
 623 00014178 878381828F8B898A          <1> dma_page:   db 87h,83h,81h,82h,8Fh,8Bh,89h,8Ah ; 03/08/2017
 624 00014180 0A0A0A0AD4D4D4D4          <1> dma_mask:   db 0Ah,0Ah,0Ah,0Ah,0D4h,0D4h,0D4h,0D4h
 625 00014188 0B0B0B0BD6D6D6D6          <1> dma_mod:    db 0Bh,0Bh,0Bh,0Bh,0D6h,0D6h,0D6h,0D6h
 626 00014190 0C0C0C0CD8D8D8D8          <1> dma_flip:   db 0Ch,0Ch,0Ch,0Ch,0D8h,0D8h,0D8h,0D8h
3088
3089                                                    ; 27/08/2014
3090                                                    scr_row:
3091 00014198 E0810B00                                      dd 0B8000h + 0A0h + 0A0h + 0A0h ; Row 3
3092                                                    scr_col:
3093 0001419C 00000000                                      dd 0
3094
3095                                                    Align 4
3096                                                        ; 15/04/2016
3097                                                        ; TRDOS 386 (TRDOS v2.0)
3098
3099                                                        ; 21/08/2014
3100                                                    ilist:
3101                                                        ;times       32 dd cpu_except ; INT 0 to INT 1Fh
3102                                                        ;
3103                                                        ; Exception list
3104                                                        ; 25/08/2014
3105 000141A0 [E20B0000]                                    dd     exc0   ; 0h,  Divide-by-zero Error
3106 000141A4 [E90B0000]                                    dd     exc1
3107 000141A8 [F00B0000]                                    dd     exc2
3108 000141AC [F70B0000]                                    dd     exc3
3109 000141B0 [FB0B0000]                                    dd     exc4
3110 000141B4 [FF0B0000]                                    dd     exc5
3111 000141B8 [030C0000]                                    dd     exc6   ; 06h,  Invalid Opcode
3112 000141BC [070C0000]                                    dd     exc7
3113 000141C0 [0B0C0000]                                    dd     exc8
3114 000141C4 [0F0C0000]                                    dd     exc9
3115 000141C8 [130C0000]                                    dd     exc10
3116 000141CC [170C0000]                                    dd     exc11
3117 000141D0 [1B0C0000]                                    dd     exc12
3118 000141D4 [1F0C0000]                                    dd     exc13  ; 0Dh, General Protection Fault
3119 000141D8 [230C0000]                                    dd     exc14  ; 0Eh, Page Fault
3120 000141DC [270C0000]                                    dd     exc15
3121 000141E0 [2B0C0000]                                    dd     exc16
3122 000141E4 [2F0C0000]                                    dd     exc17
3123 000141E8 [330C0000]                                    dd     exc18
3124 000141EC [370C0000]                                    dd     exc19
3125 000141F0 [3B0C0000]                                    dd     exc20
3126 000141F4 [3F0C0000]                                    dd     exc21
3127 000141F8 [430C0000]                                    dd     exc22
3128 000141FC [470C0000]                                    dd     exc23
3129 00014200 [4B0C0000]                                    dd     exc24
3130 00014204 [4F0C0000]                                    dd     exc25
3131 00014208 [530C0000]                                    dd     exc26
3132 0001420C [570C0000]                                    dd     exc27
3133 00014210 [5B0C0000]                                    dd     exc28
3134 00014214 [5F0C0000]                                    dd     exc29
3135 00014218 [630C0000]                                    dd     exc30
3136 0001421C [670C0000]                                    dd     exc31
3137                                                    IRQ_list: ; 28/02/2017 ('syscalbac')
3138                                                        ; Interrupt list
3139 00014220 [56090000]                                    dd     timer_int    ; INT 20h
3140                                                            ;dd    irq0
3141 00014224 [CE100000]                                    dd     kb_int       ; 24/01/2016
3142                                                            ;dd    irq1
3143 00014228 [380B0000]                                    dd     irq2
3144                                                            ; COM2 int
3145 0001422C [3C0B0000]                                    dd     irq3
3146                                                            ; COM1 int
3147 00014230 [470B0000]                                    dd     irq4
3148 00014234 [520B0000]                                    dd     irq5
3149                                                    ;DISKETTE_INT: ;06/02/2015
3150 00014238 [C9510000]                                    dd     fdc_int             ; 16/02/2015, IRQ 6 handler
3151                                                            ;dd    irq6
3152                                                    ; Default IRQ 7 handler against spurious IRQs (from master PIC)
3153                                                    ; 25/02/2015 (source: http://wiki.osdev.org/8259_PIC)
3154 0001423C [C10E0000]                                    dd     default_irq7 ; 25/02/2015
3155                                                            ;dd    irq7
3156                                                    ; Real Time Clock Interrupt
3157 00014240 [C10A0000]                                    dd     rtc_int             ; 23/02/2015, IRQ 8 handler
3158                                                            ;dd    irq8   ; INT 28h
3159 00014244 [620B0000]                                    dd     irq9
3160 00014248 [660B0000]                                    dd     irq10
3161 0001424C [6A0B0000]                                    dd     irq11
3162 00014250 [6E0B0000]                                    dd     irq12
```

```
3163 00014254 [720B0000]                    dd     irq13
3164                                  ;HDISK_INT1:  ;06/02/2015
3165 00014258 [7C5B0000]                    dd     hdc1_int      ; 21/02/2015, IRQ 14 handler
3166                                        ;dd    irq14
3167                                  ;HDISK_INT2:  ;06/02/2015
3168 0001425C [A35B0000]                    dd     hdc2_int      ; 21/02/2015, IRQ 15 handler
3169                                        ;dd    irq15  ; INT 2Fh
3170                                        ; 14/08/2015
3171                                        ;dd    sysent        ; INT 30h (system calls)
3172
3173                                        ; 15/04/2016
3174                                        ; TRDOS 386(TRDOS v2.0) Software Interrupts
3175
3176 00014260 [BD420100]                    dd     int30h        ; Reserved for
3177                                                              ; !!! Retro UNIX (RUNIX) !!!
3178                                                              ; !!! SINGLIX !!! System Calls
3179 00014264 [C1170000]                    dd     int31h        ; Video BIOS (IBM PC/AT, Int 10h)
3180 00014268 [ED0E0000]                    dd     int32h        ; Keyboard Functions (IBM PC/AT, Int 16h)
3181 0001426C [80520000]                    dd     int33h        ; DISK I/O (IBM PC/AT, Int 13h)
3182 00014270 [B8240100]                    dd     int34h        ; #IOCTL# (I/O port access support for ring 3)
3183 00014274 [346A0000]                    dd     int35h        ; Time/Date Functions (IBM PC/AT, Int 1Ah)
3184 00014278 [750D0000]                    dd     ignore_int    ; INT 36h : Timer Functions
3185 0001427C [750D0000]                    dd     ignore_int    ; INT 37h
3186 00014280 [750D0000]                    dd     ignore_int    ; INT 38h
3187 00014284 [750D0000]                    dd     ignore_int    ; INT 39h
3188 00014288 [750D0000]                    dd     ignore_int    ; INT 3Ah
3189 0001428C [750D0000]                    dd     ignore_int    ; INT 3Bh
3190 00014290 [750D0000]                    dd     ignore_int    ; INT 3Ch
3191 00014294 [750D0000]                    dd     ignore_int    ; INT 3Dh
3192 00014298 [750D0000]                    dd     ignore_int    ; INT 3Eh
3193 0001429C [750D0000]                    dd     ignore_int    ; INT 3Fh
3194 000142A0 [9AD70000]                    dd     sysent        ; INT 40h : !!! TRDOS 386 System Calls !!!
3195                                        ;dd    ignore_int
3196 000142A4 00000000                      dd     0
3197
3198                                  ; 20/08/2014
3199                                    ; /* This is the default interrupt "handler" :-) */
3200                                    ; Linux v0.12 (head.s)
3201                                  int_msg:
3202 000142A8 556E6B6E6F776E2069-           db "Unknown interrupt ! ", 0
3202 000142B1 6E7465727275707420-
3202 000142BA 212000
3203
3204                                  ; 15/04/2016
3205                                  ; TRDOS 386 (TRDOS v2.0)
3206
3207                                  ; 29/04/2016
3208                                  int30h:
3209                                  trdos_isc_routine:
3210                                        ; 02/05/2016
3211                                        ; 01/05/2016
3212                                        ; 29/04/2016
3213                                        ; 18/04/2016
3214                                        ; 15/04/2016 (TRDOS 386 = TRDOS v2.0)
3215                                        ; 17/04/2011 (TRDOS v1.0, 'IFC.ASM')
3216                                        ; 03/02/2011 ('trdos_ifc_routine')
3217                                        ;
3218 000142BD 8B1C24                      mov    ebx, [esp] ; EIP (next)
3219 000142C0 83EB02                      sub    ebx, 2 ; EIP (CD ##h)
3220
3221 000142C3 89C1                        mov    ecx, eax
3222 000142C5 8A4301                      mov    al, [ebx+1] ; CDh ##h
3223
3224 000142C8 66BA1000                    mov    dx, KDATA
3225 000142CC 8EDA                        mov    ds, dx
3226 000142CE 8EC2                        mov    es, dx
3227
3228 000142D0 FC                          cld
3229 000142D1 8B15[90810100]              mov    edx, [k_page_dir]
3230 000142D7 0F22DA                      mov    cr3, edx
3231
3232 000142DA E8E4FFFEFF                  call   bytetohex
3233 000142DF 66A3[153F0100]              mov    [int_num_str], ax
3234
3235 000142E5 89D8                        mov    eax, ebx ; EIP
3236 000142E7 E81700FFFF                  call   dwordtohex
3237 000142EC 8915[313F0100]              mov    [eip_str], edx
3238 000142F2 A3[353F0100]                mov    [eip_str+4], eax
3239
3240 000142F7 89C8                        mov    eax, ecx
3241 000142F9 E80500FFFF                  call   dwordtohex
3242 000142FE 8915[203F0100]              mov    [eax_str], edx
3243 00014304 A3[243F0100]                mov    [eax_str+4], eax
3244
3245 00014309 43                          inc    ebx
3246 0001430A 8A03                        mov    al, [ebx] ; Interrupt number
3247
3248                                  trdos_isc_handler:
3249 0001430C 80FE30                      cmp    dh, 30h ; Retro UNIX, SINGLIX System calls
3250 0001430F 7507                        jne    short trdos_usi_handler
3251 00014311 BE[B23E0100]                mov    esi, isc_msg
3252 00014316 EB05                        jmp    short loc_write_inv_system_call_msg
3253
3254                                  trdos_usi_handler:
3255 00014318 BE[C83E0100]                mov    esi, usi_msg
3256
3257                                  loc_write_inv_system_call_msg:
3258 0001431D E82332FFFF                  call   print_msg
3259                                      ; 29/04/2016
3260 00014322 BE[FE3E0100]                mov    esi, inv_msg_for_trdos_v2
3261 00014327 E81932FFFF                  call   print_msg
3262
3263                                  loc_ifc_terminate_process:
3264                                      ; u.uno = process number
3265                                      ; 29/04/2016
```

```
                                        ; 02/05/2016
0001432C FE05[5B030300]                 inc     byte [sysflg] ; 0FFh -> 0

00014332 B801000000                     mov     eax, 1
00014337 E93797FFFF                     jmp     sysexit

                                    ; 07/03/2015
                                    ; Temporary Code
                                    display_disks:
0001433C 803D[EE6D0000]00               cmp     byte [fd0_type], 0
00014343 7605                           jna     short ddsks1
00014345 E87D000000                     call    pdskm
                                    ddsks1:
0001434A 803D[EF6D0000]00               cmp     byte [fd1_type], 0
00014351 760C                           jna     short ddsks2
00014353 C605[97440100]31               mov     byte [dskx], '1'
0001435A E868000000                     call    pdskm
                                    ddsks2:
0001435F 803D[F06D0000]00               cmp     byte [hd0_type], 0
00014366 7654                           jna     short ddsk6
00014368 66C705[95440100]68-            mov     word [dsktype], 'hd'
00014370 64
00014371 C605[97440100]30               mov     byte [dskx], '0'
00014378 E84A000000                     call    pdskm
                                    ddsks3:
0001437D 803D[F16D0000]00               cmp     byte [hd1_type], 0
00014384 7636                           jna     short ddsk6
00014386 C605[97440100]31               mov     byte [dskx], '1'
0001438D E835000000                     call    pdskm
                                    ddsks4:
00014392 803D[F26D0000]00               cmp     byte [hd2_type], 0
00014399 7621                           jna     short ddsk6
0001439B C605[97440100]32               mov     byte [dskx], '2'
000143A2 E820000000                     call    pdskm
                                    ddsks5:
000143A7 803D[F36D0000]00               cmp     byte [hd3_type], 0
000143AE 760C                           jna     short ddsk6
000143B0 C605[97440100]33               mov     byte [dskx], '3'
000143B7 E80B000000                     call    pdskm
                                    ddsk6:
000143BC BE[BF440100]                   mov     esi, nextline
000143C1 E806000000                     call    pdskml
                                    pdskm_ok:
000143C6 C3                             retn
                                    pdskm:
000143C7 BE[93440100]                   mov     esi, dsk_ready_msg
                                    pdskml:
000143CC AC                             lodsb
000143CD 08C0                           or      al, al
000143CF 74F5                           jz      short pdskm_ok
000143D1 56                             push    esi
                                        ; 13/05/2016
000143D2 BB07000000                       mov     ebx, 7  ; Black background,
                                                  ; light gray forecolor
                                                  ; Video page 0 (bh=0)
000143D7 E821DFFEFF                     call    _write_tty
000143DC 5E                             pop     esi
000143DD EBED                           jmp     short pdskml

000143DF 90                         Align 2
                                        ; 21/08/2014
                                    exc_msg:
000143E0 435055206578636570-            db "CPU exception ! "
000143E9 74696F6E202120

                                    excnstr:            ; 25/08/2014
000143F0 3F3F68202045495020-            db "??h", "  EIP : "
000143F9 3A20

                                    EIPstr: ; 29/08/2014
000143FB 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:
00014407 07                             db    07h
00014408 0D0A                           db    0Dh, 0Ah
                                        ;db   "MEMORY ALLOCATION INFO", 0Dh, 0Ah, 0Dh, 0Ah
0001440A 546F74616C206D656D-            db    "Total memory : "
00014413 6F7279203A20

                                    mem_total_b_str: ; 10 digits
00014419 303030303030303030-            db    "0000000000 bytes", 0Dh, 0Ah
00014422 302062797465730D0A

0001442B 202020202020202020-            db    "               ", 20h, 20h, 20h
00014434 202020202020202020

                                    mem_total_p_str: ; 7 digits
0001443D 303030303030302070-            db    "0000000 pages", 0Dh, 0Ah
00014446 616765730D0A

0001444C 0D0A                           db    0Dh, 0Ah
0001444E 46726565206D656D6F-            db    "Free memory  : "
00014457 727920203A20

                                    free_mem_b_str:  ; 10 digits
0001445D 3F3F3F3F3F3F3F3F3F-            db    "?????????? bytes", 0Dh, 0Ah
00014466 3F2062797465730D0A

0001446F 202020202020202020-            db    "               ", 20h, 20h, 20h
00014478 202020202020202020

                                    free_mem_p_str:  ; 7 digits
00014481 3F3F3F3F3F3F3F2070-            db    "??????? pages", 0Dh, 0Ah
0001448A 616765730D0A
```

```
3360 00014490 0D0A00                      db     0Dh, 0Ah, 0
3361
3362                                  dsk_ready_msg:
3363 00014493 0D0A                        db     0Dh, 0Ah
3364                                  dsktype:
3365 00014495 6664                        db     'fd'
3366                                  dskx:
3367 00014497 30                          db     '0'
3368 00014498 20                          db     20h
3369 00014499 697320524541445920-         db     'is READY ...'
3369 000144A2 2E2E2E
3370 000144A5 00                          db     0
3371
3372                                  setup_error_msg:
3373 000144A6 0D0A                        db 0Dh, 0Ah
3374 000144A8 4469736B2053657475-         db 'Disk Setup Error !'
3374 000144B1 70204572726F722021
3375 000144BA 0D0A00                      db 0Dh, 0Ah,0
3376
3377                                  next2line: ; 08/02/2016
3378 000144BD 0D0A                        db     0Dh, 0Ah
3379                                  nextline:
3380 000144BF 0D0A00                      db     0Dh, 0Ah, 0
3381
3382                                  ; temporary
3383                                  ; 19/12/2020
3384                                  msg_lfb_addr:
3385                                       ;db    0Dh, 0Ah
3386 000144C2 4C696E656172206672-         db     "Linear frame buffer at "
3386 000144CB 616D65206275666665-
3386 000144D4 7220617420
3387                                  lfb_addr_str: ; 8 (hex) digits
3388 000144D9 303030303030303068-         db     "00000000h", 0Dh, 0Ah
3388 000144E2 0D0A
3389 000144E4 0D0A00                      db     0Dh, 0Ah, 0
3390
3391                                  ; KERNEL - SYSINIT Messages
3392                                  ; 24/08/2015
3393                                  ; 13/04/2015 - (Retro UNIX 386 v1 Beginning)
3394                                  ; 14/07/2013
3395                                  ;kernel_init_err_msg:
3396                                  ;     db 0Dh, 0Ah
3397                                  ;     db 07h
3398                                  ;     db 'Kernel initialization ERROR !'
3399                                  ;     db 0Dh, 0Ah, 0
3400
3401                                  ;welcome_msg:
3402                                  ;     db 0Dh, 0Ah
3403                                  ;     db 07h
3404                                  ;     db 'Welcome to TRDOS 386 Operating System !'
3405                                  ;     db 0Dh, 0Ah
3406                                  ;     db 'by Erdogan Tan - 31/12/2017 (v2.0.0)'
3407                                  ;     db 0Dh, 0Ah, 0
3408
3409                                  panic_msg:
3410 000144E7 0D0A07                        db 0Dh, 0Ah, 07h
3411 000144EA 4552524F523A204B65-         db 'ERROR: Kernel Panic !'
3411 000144F3 726E656C2050616E69-
3411 000144FC 632021
3412 000144FF 0D0A00                      db 0Dh, 0Ah, 0
3413
3414                                  ;msgl_drv_not_ready:
3415                                  ;     db 07h, 0Dh, 0Ah
3416                                  ;       db 'Drive not ready or read error !'
3417                                  ;       db 0Dh, 0Ah, 0
3418
3419                                  starting_msg:
3420 00014502 5475726B6973682052-         db "Turkish Rational DOS v2.0 [24/01/2021] ...", 0
3420 0001450B 6174696F6E616C2044-
3420 00014514 4F532076322E30205B-
3420 0001451D 32342F30312F323032-
3420 00014526 315D202E2E2E00
3421                                  NextLine:
3422 0001452D 0D0A00                        db 0Dh, 0Ah, 0
3423
3424                                      %include 'audio.s' ; 03/04/2017
   1                                  <1> ; ****************************************************************************
   2                                  <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.2 - audio.s
   3                                  <1> ; ----------------------------------------------------------------------------
   4                                  <1> ; Last Update: 01/09/2020
   5                                  <1> ; ----------------------------------------------------------------------------
   6                                  <1> ; Beginning: 03/04/2017
   7                                  <1> ; ----------------------------------------------------------------------------
   8                                  <1> ; Assembler: NASM version 2.11 (trdos386.s)
   9                                  <1> ; ****************************************************************************
  10                                  <1>
  11                                  <1> ; AUDIO CONTROLLER & CODEC DEFINITIONS & CODE FOR TRDOS 386
  12                                  <1>
  13                                  <1> ;=============================================================================
  14                                  <1> ;               EQUATES
  15                                  <1> ;=============================================================================
  16                                  <1>
  17                                  <1> ; PCI EQUATES
  18                                  <1>
  19                                  <1> BIT0  EQU 1
  20                                  <1> BIT1  EQU 2
  21                                  <1> BIT2  EQU 4
  22                                  <1> BIT3  EQU 8
  23                                  <1> BIT4  EQU 10h
  24                                  <1> BIT5  EQU 20h
  25                                  <1> BIT6  EQU 40h
  26                                  <1> BIT7  EQU 80h
  27                                  <1> BIT8  EQU 100h
  28                                  <1> BIT9  EQU 200h
  29                                  <1> BIT10 EQU 400h
```

```
30                                      <1> BIT11 EQU 800h
31                                      <1> BIT12 EQU 1000h
32                                      <1> BIT13 EQU 2000h
33                                      <1> BIT14 EQU 4000h
34                                      <1> BIT15 EQU 8000h
35                                      <1> BIT16 EQU 10000h
36                                      <1> BIT17 EQU 20000h
37                                      <1> BIT18 EQU 40000h
38                                      <1> BIT19 EQU 80000h
39                                      <1> BIT20 EQU 100000h
40                                      <1> BIT21 EQU 200000h
41                                      <1> BIT22 EQU 400000h
42                                      <1> BIT23 EQU 800000h
43                                      <1> BIT24 EQU 1000000h
44                                      <1> BIT25 EQU 2000000h
45                                      <1> BIT26 EQU 4000000h
46                                      <1> BIT27 EQU 8000000h
47                                      <1> BIT28 EQU 10000000h
48                                      <1> BIT29 EQU 20000000h
49                                      <1> BIT30 EQU 40000000h
50                                      <1> BIT31 EQU 80000000h
51                                      <1> NOT_BIT31 EQU 7FFFFFFFh
52                                      <1>
53                                      <1> ; PCI equates
54                                      <1> ; PCI function address (PFA)
55                                      <1> ; bit 31 = 1
56                                      <1> ; bit 23:16 = bus number     (0-255)
57                                      <1> ; bit 15:11 = device number  (0-31)
58                                      <1> ; bit 10:8 = function number (0-7)
59                                      <1> ; bit 7:0 = register number  (0-255)
60                                      <1>
61                                      <1> IO_ADDR_MASK    EQU     0FFFEh  ; mask off bit 0 for reading BARs
62                                      <1> PCI_INDEX_PORT  EQU     0CF8h
63                                      <1> PCI_DATA_PORT   EQU     0CFCh
64                                      <1> PCI32           EQU     BIT31   ; bitflag to signal 32bit access
65                                      <1> PCI16           EQU     BIT30   ; bitflag for 16bit access
66                                      <1> NOT_PCI32_PCI16    EQU   03FFFFFFFh ; NOT BIT31+BIT30 ; 19/03/2017
67                                      <1>
68                                      <1> PCI_FN0         EQU     0 << 8
69                                      <1> PCI_FN1         EQU     1 << 8
70                                      <1> PCI_FN2         EQU     2 << 8
71                                      <1> PCI_FN3         EQU     3 << 8
72                                      <1> PCI_FN4         EQU     4 << 8
73                                      <1> PCI_FN5         EQU     5 << 8
74                                      <1> PCI_FN6         EQU     6 << 8
75                                      <1> PCI_FN7         EQU     7 << 8
76                                      <1>
77                                      <1> PCI_CMD_REG  EQU    04h    ; reg 04, command reg
78                                      <1> IO_ENA              EQU    BIT0   ; i/o decode enable
79                                      <1> MEM_ENA             EQU    BIT1   ; memory decode enable
80                                      <1> BM_ENA              EQU     BIT2 ; bus master enable
81                                      <1>
82                                      <1> ; VIA VT8233 EQUATES
83                                      <1>
84                                      <1> VIA_VID             equ 1106h    ; VIA's PCI vendor ID
85                                      <1> VT8233_DID      equ 3059h ; VT8233 (VT8235) device ID
86                                      <1>
87                                      <1> PCI_IO_BASE          equ 10h
88                                      <1> AC97_INT_LINE        equ 3Ch
89                                      <1> VIA_ACLINK_CTRL      equ 41h
90                                      <1> VIA_ACLINK_STAT      equ 40h
91                                      <1> VIA_ACLINK_C00_READY equ 01h ; primary codec ready
92                                      <1>
93                                      <1> VIA_REG_AC97     equ 80h ; dword
94                                      <1>
95                                      <1> VIA_ACLINK_CTRL_ENABLE    equ   80h ; 0: disable, 1: enable
96                                      <1> VIA_ACLINK_CTRL_RESET     equ   40h ; 0: assert, 1: de-assert
97                                      <1> VIA_ACLINK_CTRL_SYNC      equ   20h ; 0: release SYNC, 1: force SYNC hi
98                                      <1> VIA_ACLINK_CTRL_VRA       equ   08h ; 0: disable VRA, 1: enable VRA
99                                      <1> VIA_ACLINK_CTRL_PCM       equ   04h ; 0: disable PCM, 1: enable PCM
100                                     <1>                                     ; 3D Audio Channel slots 3/4
104                                     <1> VIA_ACLINK_CTRL_INIT      equ  (VIA_ACLINK_CTRL_ENABLE +
VIA_ACLINK_CTRL_RESET +                         VIA_ACLINK_CTRL_PCM +                                   VIA_ACLINK_CTRL_VRA)
105                                     <1>
106                                     <1> CODEC_AUX_VOL             equ   04h
107                                     <1> VIA_REG_AC97_BUSY  equ   01000000h ;(1<<24)
108                                     <1> VIA_REG_AC97_CMD_SHIFT    equ   10h ; 16
109                                     <1> VIA_REG_AC97_PRIMARY_VALID equ 02000000h ;(1<<25)
110                                     <1> VIA_REG_AC97_READ  equ   00800000h ;(1<<23)
111                                     <1> VIA_REG_AC97_CODEC_ID_SHIFT   equ  1Eh ; 30
112                                     <1> VIA_REG_AC97_CODEC_ID_PRIMARY equ  0
113                                     <1> VIA_REG_AC97_DATA_SHIFT equ   0
114                                     <1> VIADEV_PLAYBACK         equ   0
115                                     <1> VIA_REG_OFFSET_STATUS   equ   0    ;; byte - channel status
116                                     <1> VIA_REG_OFFSET_CONTROL  equ   01h  ;; byte - channel control
117                                     <1> VIA_REG_CTRL_START equ   80h  ;; WO
118                                     <1> VIA_REG_CTRL_TERMINATE  equ   40h  ;; WO
119                                     <1> VIA_REG_CTRL_PAUSE      equ   08h  ;; RW
120                                     <1> VIA_REG_CTRL_RESET      equ   01h  ;; RW - probably reset? undocumented
121                                     <1> VIA_REG_OFFSET_STOP_IDX equ   08h  ;; dword - stop index, channel type, sample rate
122                                     <1> VIA8233_REG_TYPE_16BIT  equ   200000h ;; RW
123                                     <1> VIA8233_REG_TYPE_STEREO equ   100000h ;; RW
124                                     <1> VIA_REG_OFFSET_CURR_INDEX equ 0Fh ;; byte - channel current index (for via8233 only)
125                                     <1> VIA_REG_OFFSET_TABLE_PTR equ  04h  ;; dword - channel table pointer
126                                     <1> VIA_REG_OFFSET_CURR_PTR equ   04h  ;; dword - channel current pointer
127                                     <1> VIA_REG_OFS_PLAYBACK_VOLUME_L equ  02h ;; byte
128                                     <1> VIA_REG_OFS_PLAYBACK_VOLUME_R equ  03h ;; byte
129                                     <1> VIA_REG_CTRL_AUTOSTART    equ   20h
130                                     <1> VIA_REG_CTRL_INT_EOL      equ   02h
131                                     <1> VIA_REG_CTRL_INT_FLAG     equ   01h
134                                     <1> VIA_REG_CTRL_INT   equ  (VIA_REG_CTRL_INT_FLAG +                                    VIA_REG_CTRL_INT_EOL
+                               VIA_REG_CTRL_AUTOSTART)
135                                     <1>
136                                     <1> VIA_REG_STAT_STOP_IDX     equ   10h    ;; RO ; 27/07/2020
137                                     <1>                                  ; current index = stop index
```

```
138                             <1> VIA_REG_STAT_STOPPED      equ   04h    ;; RWC
139                             <1> VIA_REG_STAT_EOL   equ   02h    ;; RWC
140                             <1> VIA_REG_STAT_FLAG  equ   01h    ;; RWC
141                             <1> VIA_REG_STAT_ACTIVE       equ   80h    ;; RO
142                             <1> ; 28/11/2016
143                             <1> VIA_REG_STAT_LAST  equ   40h    ;; RO
144                             <1> VIA_REG_STAT_TRIGGER_QUEUED equ 08h  ;; RO
145                             <1> VIA_REG_CTRL_INT_STOP     equ   04h  ; Interrupt on Current Index = Stop Index
146                             <1>                              ; and End of Block
147                             <1>
148                             <1> VIA_REG_OFFSET_CURR_COUNT equ 0Ch ;; dword - channel current count, index
149                             <1>
150                             <1> PORTB       EQU    061h
151                             <1> REFRESH_STATUS     EQU    010h   ; Refresh signal status
152                             <1>
153                             <1> ; AC97 Codec registers.
154                             <1>
155                             <1> ; 22/07/2020
156                             <1> ; REALTEK ALC655 and ADI SOUNDMAX AD1980 CODEC MIXER REGISTERS
157                             <1>
158                             <1> ; each codec/mixer register is 16bits
159                             <1>
160                             <1> CODEC_RESET_REG                 equ     00h  ; reset codec
161                             <1> CODEC_MASTER_VOL_REG            equ     02h  ; master volume
162                             <1> CODEC_HP_VOL_REG                equ     04h  ; headphone volume ; AD1980
163                             <1> CODEC_MASTER_MONO_VOL_REG       equ     06h  ; master mono volume (mono-out)
164                             <1> ;CODEC_MASTER_TONE_REG          equ     08h  ; master tone (R+L) ; (not used)
165                             <1> CODEC_PCBEEP_VOL_REG            equ     0Ah  ; PC beep volume ; ALC655
166                             <1> CODEC_PHONE_VOL_REG             equ     0Ch  ; phone volume
167                             <1> CODEC_MIC_VOL_REG               equ     0Eh  ; mic volume
168                             <1> CODEC_LINE_IN_VOL_REG           equ     10h  ; line in volume
169                             <1> CODEC_CD_VOL_REG                equ     12h  ; CD volume
170                             <1> ;CODEC_VID_VOL_REG              equ     14h  ; video volume ; (not used)
171                             <1> CODEC_AUX_VOL_REG               equ     16h  ; aux volume
172                             <1> CODEC_PCM_OUT_REG               equ     18h  ; PCM out volume
173                             <1> CODEC_RECORD_SELECT_REG         equ     1Ah  ; record select
174                             <1> CODEC_RECORD_VOL_REG            equ     1Ch  ; record volume (record gain)
175                             <1> ;CODEC_RECORD_MIC_VOL_REG       equ     1Eh  ; record mic volume ; (not used)
176                             <1> CODEC_GP_REG                    equ     20h  ; general purpose
177                             <1> ;CODEC_3D_CONTROL_REG           equ     22h  ; 3D control
178                             <1> ;;CODEC_AUDIO_INT_PAGING_REG    equ    24h   ; audio int & paging ; (not used)
179                             <1> CODEC_POWER_CTRL_REG            equ     26h  ; power down control
180                             <1> CODEC_EXT_AUDIO_REG             equ     28h  ; extended audio ID
181                             <1> CODEC_EXT_AUDIO_CTRL_REG        equ     2Ah  ; extended audio status/control
182                             <1> CODEC_PCM_FRONT_DACRATE_REG     equ     2Ch  ; PCM front sample rate
183                             <1> CODEC_PCM_SURND_DACRATE_REG     equ     2Eh  ; PCM surround sample rate
184                             <1> CODEC_PCM_LFE_DACRATE_REG       equ     30h  ; PCM Center/LFE sample rate
185                             <1> CODEC_LR_ADCRATE_REG            equ     32h  ; PCM input sample rate
186                             <1> CODEC_MIC_ADCRATE_REG           equ     34h  ; mic in sample rate  ; AD1980
187                             <1> CODEC_PCM_LFE_VOL_REG           equ     36h  ; PCM Center/LFE volume
188                             <1> CODEC_PCM_SURND_VOL_REG         equ     38h  ; PCM surround volume
189                             <1> ;CODEC_SPDIF_CTRL_REG           equ     3Ah  ; S/PDIF control
190                             <1> ; 22/07/2020
191                             <1> CODEC_MISC_CRTL_BITS_REG  equ    76h    ; misc control bits ; AD1980
192                             <1> ;
193                             <1> CODEC_VENDOR_ID1          equ    7Ch    ; REALTEK: 414Ch, ADI: 4144h
194                             <1> CODEC_VENDOR_ID2          equ    7Eh    ; REALTEK: 4760h, ADI: 5370h
195                             <1>
196                             <1> ; VT8233 SGD bits (21/04/2017)
197                             <1> FLAG  EQU BIT30
198                             <1> EOL   EQU BIT31
199                             <1>
200                             <1> ; INTEL ICH EQUATES
201                             <1> ; 28/05/2017
202                             <1> INTEL_VID   equ    8086h  ; Intel's PCI vendor ID
203                             <1> ICH_DID           equ    2415h  ; ICH (82801AA) device ID
204                             <1> NAMBAR_REG      equ      10h    ; native audio mixer Base Address Register
205                             <1> NABMBAR_REG     equ      14h    ; native audio bus mastering Base Addr Reg
206                             <1>
207                             <1> PI_CR_REG       equ     0Bh     ; PCM in Control Register
208                             <1> PO_CR_REG   equ     1Bh     ; PCM out Control Register
209                             <1> MC_CR_REG   equ     2Bh     ; MIC in Control Register
210                             <1>
211                             <1> PI_SR_REG   equ     6       ; PCM in Status register
212                             <1> PO_SR_REG   equ     16h     ; PCM out Status register
213                             <1> MC_SR_REG   equ     26h     ; MIC in Status register
214                             <1>
215                             <1> IOCE        equ     BIT4    ; interrupt on complete enable.
216                             <1> FEIFE       equ     BIT3    ; set if you want an interrupt to fire
217                             <1> LVBIE       equ     BIT2    ; last valid buffer interrupt enable.
218                             <1> RR          equ     BIT1    ; reset registers. Nukes all regs
219                             <1>                                ; except bits 4:2 of this register.
220                             <1>                                ; Only set this bit if BIT 0 is 0
221                             <1> RPBM        equ     BIT0    ; Run/Pause
222                             <1>                             ; set this bit to start the codec!
223                             <1>
224                             <1> PI_BDBAR_REG equ     0       ; PCM in buffer descriptor BAR
225                             <1> PO_BDBAR_REG equ     10h     ; PCM out buffer descriptor BAR
226                             <1> MC_BDBAR_REG equ     20h     ; MIC in buffer descriptor BAR
227                             <1>
228                             <1> PI_CIV_REG  equ     4       ; PCM in current Index value (RO)
229                             <1> PO_CIV_REG  equ     14h     ; PCM out current Index value (RO)
230                             <1> MC_CIV_REG  equ     24h     ; MIC in current Index value (RO)
231                             <1>
232                             <1> PI_LVI_REG  equ     5       ; PCM in Last Valid Index
233                             <1> PO_LVI_REG  equ     15h     ; PCM out Last Valid Index
234                             <1> MC_LVI_REG  equ     25h     ; MIC in Last Valid Index
235                             <1>
236                             <1> IOC         equ     BIT31; Fire an interrupt whenever this
237                             <1>                             ; buffer is complete.
238                             <1> BUP         equ     BIT30; Buffer Underrun Policy.
239                             <1>
240                             <1> GLOB_CNT_REG equ     2Ch     ; Global Control Register
241                             <1> GLOB_STS_REG equ     30h     ; Global Status register (RO)
242                             <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
00014530 B886801524     <1>       mov     eax, (ICH_DID << 16) + INTEL_VID
00014535 E876000000     <1>         call    pciFindDevice
0001453A 730D           <1>         jnc     short d_ac97_1
                        <1> d_ac97_0:
                        <1> ; couldn't find the audio device!
0001453C 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
0001453D B806115930     <1>       mov     eax, (VT8233_DID << 16) + VIA_VID
00014542 E869000000     <1>         call    pciFindDevice
                        <1> ;       jnc     short d_vt8233_0
                        <1> ; couldn't find the audio device!
                        <1> ;     retn
00014547 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>
00014549 A3[18940100]   <1>       mov   [audio_dev_id], eax
0001454E 8915[1C940100] <1>       mov   [audio_vendor], edx
                        <1>
                        <1>       ; init controller
00014554 B004           <1>       mov   al, PCI_CMD_REG ; command register (04h)
00014556 E8E2000000     <1>       call  pciRegRead32
                        <1>
                        <1>       ; eax = BUS/DEV/FN/REG
                        <1>       ; edx = STATUS/COMMAND
                        <1>       ;     SSSSSSSSSSSSSSSSCCCCCCCCCCCCCCCC
0001455B 8915[20940100] <1>       mov   [audio_stats_cmd], edx
                        <1>
00014561 B010           <1>       mov   al, PCI_IO_BASE ; IO base address register (10h)
                        <1>       ;mov  al, NAMBAR_REG      ; Native Audio Mixer BAR (10h)
00014563 E8D5000000     <1>       call  pciRegRead32
                        <1>
00014568 66813D[1C940100]86- <1>  cmp   word [audio_vendor], 8086h ; AC'97 ?
00014570 80             <1>
00014571 751F           <1>       jne   short d_vt8233_1
                        <1>
00014573 6683E2FE       <1>       and   dx, 0FFFEh ; Audio Codec IO_ADDR_MASK
00014577 668915[48940100] <1>     mov   [NAMBAR], dx
                        <1>
0001457E B014           <1>       mov   al, NABMBAR_REG ; Native Audio Bus Mastering BAR (14h)
00014580 E8B8000000     <1>       call  pciRegRead32
                        <1>
00014585 6683E2C0       <1>       and   dx, 0FFC0h ; Audio Controller IO_ADDR_MASK
00014589 668915[4A940100] <1>     mov   [NABMBAR], dx
                        <1>         ;mov [audio_io_base], dx
                        <1>
00014590 EB0B           <1>       jmp   short d_ac97_2
                        <1>
                        <1> d_vt8233_1:
00014592 6683E2C0       <1>       and     dx, 0FFC0h ; Audio Controller IO_ADDR_MASK
00014596 668915[16940100] <1>       mov     [audio_io_base], dx
                        <1>
                        <1> d_ac97_2:
                        <1>       ; 10/06/2017
0001459D B03C           <1>       mov   al, AC97_INT_LINE ; Interrupt Line Register (3Ch)
                        <1>       ;call pciRegRead32
0001459F E886000000     <1>       call  pciRegRead8
                        <1>
                        <1>       ;and  edx, 0FFh
000145A4 6681E2FF00     <1>       and   dx, 0FFh
                        <1>
000145A9 8815[13940100] <1>       mov     [audio_intr], dl
                        <1>
000145AF 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
000145B0 50                     <1>       push   eax
                                <1>       ;push  esi
                                <1>       ;push  edi
                                <1>
000145B1 89C6                   <1>         mov     esi, eax                ; save off vend+device ID
000145B3 BF00FFFF7F             <1>         mov     edi, (80000000h - 100h) ; start with bus 0, dev 0 func 0
                                <1>
                                <1> nextPCIdevice:
000145B8 81C700010000           <1>         add     edi, 100h
000145BE 81FF00F8FF80           <1>         cmp     edi, 80FFF800h         ; scanned all devices?
000145C4 F9                     <1>         stc
000145C5 740C                   <1>         je      short PCIScanExit       ; not found
                                <1>
000145C7 89F8                   <1>         mov     eax, edi                ; read PCI registers
000145C9 E86F000000             <1>         call    pciRegRead32
000145CE 39F2                   <1>         cmp     edx, esi                ; found device?
000145D0 75E6                   <1>         jne     short nextPCIdevice
000145D2 F8                     <1>         clc
                                <1>
                                <1> PCIScanExit:
000145D3 9C                     <1>       pushf
000145D4 B8FFFFFF7F             <1>       mov    eax, NOT_BIT31      ; 19/03/2017
000145D9 21F8                   <1>       and    eax, edi     ; return only bus/dev/fn #
000145DB 9D                     <1>       popf
                                <1>
                                <1>       ;pop   edi
                                <1>       ;pop   esi
000145DC 5A                     <1>       pop    edx
                                <1>       ;pop   ecx
000145DD 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>
000145DE 53                     <1>       push   ebx
000145DF 51                     <1>       push   ecx
000145E0 89C3                   <1>         mov     ebx, eax        ; save eax, dh
000145E2 88F1                   <1>         mov     cl, dh
                                <1>
000145E4 25FFFFFF3F             <1>         and     eax, NOT_PCI32_PCI16   ; clear out data size request
000145E9 0D00000080             <1>         or      eax, BIT31             ; make a PCI access request
000145EE 24FC                   <1>         and     al, ~3 ; NOT 3         ; force index to be dword
                                <1>
000145F0 66BAF80C               <1>         mov     dx, PCI_INDEX_PORT
000145F4 EF                     <1>         out  dx, eax                   ; write PCI selector
                                <1>
000145F5 66BAFC0C               <1>         mov     dx, PCI_DATA_PORT
000145F9 88D8                   <1>         mov     al, bl
000145FB 2403                   <1>         and     al, 3                  ; figure out which port to
000145FD 00C2                   <1>         add     dl, al                 ; read to
                                <1>
000145FF F7C3000000C0           <1>       test    ebx, PCI32+PCI16
00014605 7507                   <1>         jnz     short _pregr0
00014607 EC                     <1>       in     al, dx             ; return 8 bits of data
00014608 88C2                   <1>         mov  dl, al
0001460A 88CE                   <1>       mov     dh, cl                   ; restore dh for 8 bit read
0001460C EB12                   <1>       jmp    short _pregr2
                                <1> _pregr0:
0001460E F7C300000080           <1>       test    ebx, PCI32
00014614 7507                   <1>         jnz  short _pregr1
00014616 66ED                   <1>       in     ax, dx
00014618 6689C2                 <1>         mov     dx, ax                 ; return 16 bits of data
0001461B EB03                   <1>       jmp    short _pregr2
                                <1> _pregr1:
0001461D ED                     <1>       in     eax, dx                  ; return 32 bits of data
0001461E 89C2                   <1>       mov    edx, eax
                                <1> _pregr2:
00014620 89D8                   <1>       mov     eax, ebx          ; restore eax
00014622 25FFFFFF3F             <1>         and     eax, NOT_PCI32_PCI16   ; clear out data size request
00014627 59                     <1>       pop    ecx
00014628 5B                     <1>       pop    ebx
00014629 C3                     <1>       retn
                                <1>
                                <1> pciRegRead8:
0001462A 25FFFFFF3F             <1>         and     eax, NOT_PCI32_PCI16   ; set up 8 bit read size
0001462F EBAD                   <1>         jmp     short pciRegRead; call generic PCI access
                                <1>
                                <1> pciRegRead16:
00014631 25FFFFFF3F             <1>         and     eax, NOT_PCI32_PCI16   ; set up 16 bit read size
00014636 0D00000040             <1>         or      eax, PCI16             ; call generic PCI access
0001463B EBA1                   <1>         jmp     short pciRegRead
                                <1>
                                <1> pciRegRead32:
0001463D 25FFFFFF3F             <1>         and     eax, NOT_PCI32_PCI16   ; set up 32 bit read size
00014642 0D00000080             <1>         or      eax, PCI32             ; call generic PCI access
00014647 EB95                   <1>         jmp     pciRegRead
                                <1>
                                <1> pciRegWrite:
                                <1>       ; 03/04/2017 ('pci.asm', 29/11/2016)
                                <1>       ;
                                <1>       ; 8/16/32bit PCI writer
                                <1>       ;
                                <1>       ; Entry: EAX=PCI Bus/Device/fn/register number
                                <1>       ;           BIT31 set if 32 bit access requested
                                <1>       ;           BIT30 set if 16 bit access requested
                                <1>       ;           otherwise defaults to 8bit read
                                <1>       ;         DL/DX/EDX data to write depending on size
                                <1>       ;
                                <1>       ; Note1: this routine is meant to be called via pciRegWrite8,
                                <1>       ;       pciRegWrite16 or pciRegWrite32 as detailed below.
                                <1>       ;
                                <1>       ; Note2: don't attempt to write 32bits of data from a non dword
                                <1>       ;       aligned reg number. Likewise, don't do 16 bit writes from
                                <1>       ;       non word aligned reg #
                                <1>
00014649 53                     <1>       push   ebx
0001464A 51                     <1>       push   ecx
0001464B 89C3                   <1>         mov     ebx, eax        ; save eax, edx
0001464D 89D1                   <1>         mov     ecx, edx
0001464F 25FFFFFF3F             <1>       and     eax, NOT_PCI32_PCI16     ; clear out data size request
00014654 0D00000080             <1>         or      eax, BIT31             ; make a PCI access request
00014659 24FC                   <1>         and     al, ~3 ; NOT 3         ; force index to be dword
                                <1>
0001465B 66BAF80C               <1>         mov     dx, PCI_INDEX_PORT
0001465F EF                     <1>         out  dx, eax                   ; write PCI selector
                                <1>
00014660 66BAFC0C               <1>         mov     dx, PCI_DATA_PORT
00014664 88D8                   <1>         mov     al, bl
00014666 2403                   <1>         and     al, 3                  ; figure out which port to
```

```
00014668 00C2                <1>         add     dl, al                ; write to
                             <1>
0001466A F7C3000000C0        <1>       test    ebx, PCI32+PCI16
00014670 7505                <1>         jnz     short _pregw0
00014672 88C8                <1>       mov    al, cl                   ; put data into al
00014674 EE                  <1>       out    dx, al
00014675 EB12                <1>       jmp    short _pregw2
                             <1> _pregw0:
00014677 F7C300000080        <1>       test    ebx, PCI32
0001467D 7507                <1>         jnz     short _pregw1
0001467F 6689C8              <1>       mov    ax, cx             ; put data into ax
00014682 66EF                <1>       out    dx, ax
00014684 EB03                <1>       jmp    short _pregw2
                             <1> _pregw1:
00014686 89C8                <1>       mov    eax, ecx           ; put data into eax
00014688 EF                  <1>       out    dx, eax
                             <1> _pregw2:
00014689 89D8                <1>         mov     eax, ebx        ; restore eax
0001468B 25FFFFFF3F          <1>         and     eax, NOT_PCI32_PCI16   ; clear out data size request
00014690 89CA                <1>         mov     edx, ecx        ; restore dx
00014692 59                  <1>       pop    ecx
00014693 5B                  <1>       pop    ebx
00014694 C3                  <1>       retn
                             <1>
                             <1> pciRegWrite8:
00014695 25FFFFFF3F          <1>         and     eax, NOT_PCI32_PCI16   ; set up 8 bit write size
0001469A EBAD                <1>         jmp  short pciRegWrite   ; call generic PCI access
                             <1>
                             <1> pciRegWrite16:
0001469C 25FFFFFF3F          <1>         and     eax, NOT_PCI32_PCI16   ; set up 16 bit write size
000146A1 0D00000040          <1>         or      eax, PCI16             ; call generic PCI access
000146A6 EBA1                <1>         jmp  short pciRegWrite
                             <1>
                             <1> pciRegWrite32:
000146A8 25FFFFFF3F          <1>         and     eax, NOT_PCI32_PCI16   ; set up 32 bit write size
000146AD 0D00000080          <1>         or      eax, PCI32             ; call generic PCI access
000146B2 EB95                <1>         jmp  pciRegWrite
                             <1>
                             <1> init_codec:
                             <1>       ; 05/06/2017
                             <1>       ; 28/05/2017 - Erdogan Tan (Ref: KolibriOS, vt823x.asm)
                             <1>       ;
000146B4 A1[18940100]        <1>       mov    eax, [audio_dev_id]
000146B9 B041                <1>       mov    al, VIA_ACLINK_CTRL
000146BB E86AFFFFFF          <1>       call   pciRegRead8
                             <1>       ; ?
000146C0 B040                <1>       mov    al, VIA_ACLINK_STAT
000146C2 E863FFFFFF          <1>       call   pciRegRead8
000146C7 F6C201              <1>       test   dl, VIA_ACLINK_C00_READY
000146CA 7508                <1>         jnz     short _codec_ready_1
000146CC E80E000000          <1>       call   reset_codec
000146D1 7306                <1>       jnc    short _codec_ready_2 ; eax = 1
000146D3 C3                  <1>       retn
                             <1> _codec_ready_1:
000146D4 B801000000          <1>       mov    eax, 1
                             <1> _codec_ready_2:
000146D9 E886000000          <1>       call   codec_io_w16
                             <1> detect_codec:
000146DE C3                  <1>       retn
                             <1>
                             <1> reset_codec:
                             <1>       ; 16/04/2017
                             <1>       ; 23/03/2017
                             <1>       ; ('codec.asm')
                             <1>       ; 12/11/2016 - Erdogan Tan (Ref: KolibriOS, vt823x.asm)
000146DF A1[18940100]        <1>       mov    eax, [audio_dev_id]
000146E4 B041                <1>       mov    al, VIA_ACLINK_CTRL
000146E6 B2E0                <1>              mov    dl, VIA_ACLINK_CTRL_ENABLE + VIA_ACLINK_CTRL_RESET + VIA_ACLINK_CTRL_SYNC
000146E8 E8A8FFFFFF          <1>       call   pciRegWrite8
                             <1>
000146ED E849000000          <1>       call   delay_100ms  ; wait 100 ms
                             <1>  rc_cold:
000146F2 E814000000          <1>         call    cold_reset
000146F7 7301                <1>         jnc     short _reset_codec_ok
                             <1>
                             <1>       ; 16/04/2017
                             <1>         ;xor     eax, eax         ; timeout error
                             <1>             ;stc
000146F9 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>
000146FA 29C0                <1>       sub    eax, eax
000146FC BA00000000          <1>       mov    edx, CODEC_RESET_REG ; 00h ; Reset register
00014701 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>
00014706 31C0                <1>         xor     eax, eax
                             <1>         ;mov al, VIA_ACLINK_C00_READY ; 1
00014708 FEC0                <1>         inc  al
0001470A 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
0001470B 30D2           <1>       xor    dl, dl ; 0
0001470D E883FFFFFF     <1>       call   pciRegWrite8
                        <1>
00014712 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
00014717 B2CC           <1>       mov    dl, VIA_ACLINK_CTRL_INIT
00014719 E877FFFFFF     <1>       call   pciRegWrite8
                        <1>
0001471E B910000000     <1>       mov    ecx, 16      ; total 2s
                        <1>
                        <1> _crst_wait:
                        <1>       ;mov   eax, [audio_dev_id]
00014723 B040           <1>       mov    al, VIA_ACLINK_STAT
00014725 E800FFFFFF     <1>       call   pciRegRead8
                        <1>
0001472A F6C201         <1>         test    dl, VIA_ACLINK_C00_READY
0001472D 750B           <1>         jnz     short _crst_ok
                        <1>
0001472F 51             <1>       push   ecx
00014730 E806000000     <1>       call   delay_100ms
00014735 59             <1>       pop    ecx
                        <1>
00014736 49             <1>         dec     ecx
00014737 75EA           <1>         jnz     short _crst_wait
                        <1>
                        <1> _crst_fail:
00014739 F9             <1>         stc
                        <1> _crst_ok:
0001473A C3             <1>       retn
                        <1>
                        <1> delay_100ms:
                        <1>       ; 29/05/2017
                        <1>       ; 24/03/2017 ('codec.asm')
                        <1>       ; wait 100 ms
0001473B B990010000     <1>       mov    ecx, 400  ; 400*0.25ms
                        <1> _delay_x_ms:
00014740 E803000000     <1>       call   delay1_4ms
00014745 E2F9           <1>         loop _delay_x_ms
00014747 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:
00014748 50             <1>         push    eax
00014749 51             <1>         push    ecx
0001474A B110           <1>         mov  cl, 16       ; close enough.
                        <1>
0001474C E461           <1>       in     al, PORTB ; 61h
                        <1>
0001474E 2410           <1>       and    al, REFRESH_STATUS ; 10h
00014750 88C5           <1>       mov    ch, al       ; Start toggle state
                        <1> _d4ms1:
00014752 E461           <1>       in     al, PORTB    ; Read system control port
                        <1>
00014754 2410           <1>       and    al, REFRESH_STATUS ; Refresh toggles 15.085 microseconds
00014756 38C5           <1>       cmp    ch, al
00014758 74F8           <1>       je     short _d4ms1 ; Wait for state change
                        <1>
0001475A 88C5           <1>       mov    ch, al       ; Update with new state
0001475C FEC9           <1>       dec    cl
0001475E 75F2           <1>       jnz    short _d4ms1
                        <1>
00014760 F8             <1>       clc    ; 29/05/2017
                        <1>
00014761 59             <1>         pop     ecx
00014762 58             <1>         pop     eax
00014763 C3             <1>         retn
                        <1>
                        <1> ; 10/04/2017 (TRDOS 386)
                        <1> ; 12/11/2016
                        <1>
                        <1> codec_io_w16: ;w32
                        <1>       ; ('codec.asm')
00014764 668B15[16940100]   <1>         mov  dx, [audio_io_base]
0001476B 6681C28000     <1>         add     dx, VIA_REG_AC97
00014770 EF             <1>       out    dx, eax
00014771 C3             <1>         retn
                        <1>
                        <1> codec_io_r16: ;r32
```

```
                                <1>       ; ('codec.asm')
00014772 668B15[16940100]       <1>         mov     dx, [audio_io_base]
00014779 6681C28000             <1>         add     dx, VIA_REG_AC97
0001477E ED                     <1>         in   eax, dx
0001477F C3                     <1>         retn
                                <1>
                                <1> ctrl_io_w8:
                                <1>       ; ('codec.asm')
00014780 660315[16940100]       <1>         add     dx, [audio_io_base]
00014787 EE                     <1>         out  dx, al
00014788 C3                     <1>         retn
                                <1>
                                <1> ctrl_io_r8:
                                <1>       ; ('codec.asm')
00014789 660315[16940100]       <1>         add     dx, [audio_io_base]
00014790 EC                     <1>         in   al, dx
00014791 C3                     <1>         retn
                                <1>
                                <1> ctrl_io_w32:
                                <1>       ; ('codec.asm')
00014792 660315[16940100]       <1>         add     dx, [audio_io_base]
00014799 EF                     <1>         out  dx, eax
0001479A C3                     <1>         retn
                                <1>
                                <1> ctrl_io_r32:
                                <1>       ; ('codec.asm')
0001479B 660315[16940100]       <1>         add  dx, [audio_io_base]
000147A2 ED                     <1>       in     eax, dx
000147A3 C3                     <1>         retn
                                <1>
                                <1> codec_read:
                                <1>       ; 12/11/2016 - Erdogan Tan (Ref: KolibriOS, vt823x.asm)
                                <1>         ; Use only primary codec.
                                <1>
000147A4 C1E010                 <1>         shl     eax, VIA_REG_AC97_CMD_SHIFT
000147A7 0D00008002             <1>         or      eax, VIA_REG_AC97_PRIMARY_VALID + VIA_REG_AC97_READ
                                <1>
000147AC E8B3FFFFFF             <1>       call    codec_io_w16
                                <1>
                                <1>             ; codec_valid
000147B1 E831000000             <1>       call   codec_check_ready
000147B6 7301                   <1>         jnc  short _cr_ok
                                <1>
000147B8 C3                     <1>       retn
                                <1>
                                <1> _cr_ok:
                                <1>       ; wait 25 ms
000147B9 B950000000             <1>       mov    ecx, 80 ; (100*0.25 ms)
                                <1> _cr_wloop:
000147BE E885FFFFFF             <1>       call   delay1_4ms
000147C3 E2F9                   <1>       loop   _cr_wloop
                                <1>
000147C5 E8A8FFFFFF             <1>         call    codec_io_r16
000147CA 25FFFF0000             <1>         and     eax, 0FFFFh
000147CF 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>
000147D0 C1E210                 <1>       shl     edx, VIA_REG_AC97_CMD_SHIFT
                                <1>
000147D3 C1E000                 <1>         shl     eax, VIA_REG_AC97_DATA_SHIFT ; shl eax, 0
000147D6 09C2                   <1>         or      edx, eax
                                <1>
000147D8 B800000000             <1>         mov     eax, VIA_REG_AC97_CODEC_ID_PRIMARY
000147DD C1E01E                 <1>         shl     eax, VIA_REG_AC97_CODEC_ID_SHIFT
000147E0 09D0                   <1>         or      eax, edx
                                <1>
000147E2 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:
000147E7 B914000000             <1>       mov    ecx, 20      ; total 2s
                                <1> _ccr_wait:
000147EC 51                     <1>       push   ecx
                                <1>
000147ED E880FFFFFF             <1>         call    codec_io_r16
000147F2 A900000001             <1>         test    eax, VIA_REG_AC97_BUSY
000147F7 740B                   <1>         jz      short _ccr_ok
                                <1>
000147F9 E83DFFFFFF             <1>       call   delay_100ms
                                <1>
000147FE 59                     <1>       pop    ecx
                                <1>
000147FF 49                     <1>       dec     ecx
00014800 75EA                   <1>         jnz     short _ccr_wait
                                <1>
00014802 F9                     <1>         stc
00014803 C3                     <1>         retn
                                <1>
                                <1> _ccr_ok:
00014804 59                     <1>       pop    ecx
00014805 25FFFF0000             <1>       and     eax, 0FFFFh
0001480A 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>
0001480B B802020000     <1>       mov     eax, 0202h
00014810 66A3[46940100] <1>       mov    [audio_master_volume], ax
00014816 66B81F1F       <1>       mov    ax, 1F1Fh ; 31,31
0001481A BA02000000     <1>       mov    edx, CODEC_MASTER_VOL_REG ; 02h ; Line Out
0001481F E8ACFFFFFF     <1>       call   codec_write
                        <1>       ;jc    short cconfig_error
                        <1>
                        <1>       ;mov    eax, 0202h
00014824 66B80202       <1>       mov     ax, 0202h
00014828 BA18000000     <1>       mov    edx, CODEC_PCM_OUT_REG ; 18h ; Wave Output (Stereo)
0001482D E89EFFFFFF     <1>       call   codec_write
                        <1>       ;jc    short cconfig_error
                        <1>
                        <1>       ;mov    eax, 0202h
00014832 66B80202       <1>       mov    ax, 0202h
00014836 BA04000000     <1>       mov    edx, CODEC_AUX_VOL ; 04h ; CODEC_HP_VOL_REG ; HeadPhone
0001483B E890FFFFFF     <1>       call   codec_write
                        <1>       ;jc    short cconfig_error
                        <1>
                        <1>       ;mov    eax, 08h
                        <1>         ;mov    ax,  08h
00014840 66B80880       <1>       mov    ax, 8008h ; Mute
00014844 BA0C000000     <1>       mov    edx, 0Ch  ; AC97_PHONE_VOL ; TAD Input (Mono)
00014849 E882FFFFFF     <1>       call   codec_write
                        <1>       ;jc    short cconfig_error
                        <1>
                        <1>       ;mov    eax, 0808h
0001484E 66B80808       <1>         mov     ax,  0808h
00014852 BA10000000     <1>         mov  edx, CODEC_LINE_IN_VOL_REG ; 10h ; Line Input (Stereo)
00014857 E874FFFFFF     <1>       call   codec_write
                        <1>       ;jc    short cconfig_error
                        <1>
                        <1>       ;mov    eax, 0808h
0001485C 66B80808       <1>       mov     ax,  0808h
00014860 BA12000000     <1>         mov  edx, CODEC_CD_VOL_REG ; 12h ; CR Input (Stereo)
00014865 E866FFFFFF     <1>       call   codec_write
                        <1>       ;jc    short cconfig_error
                        <1>
                        <1>       ;mov    eax, 0808h
0001486A 66B80808       <1>       mov     ax,  0808h
0001486E BA16000000     <1>         mov  edx, CODEC_AUX_VOL_REG ; 16h ; Aux Input (Stereo)
                        <1>       ;call  codec_write
                        <1>       ;;jc   short cconfig_error
00014873 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>
00014878 66BA0000       <1>         mov     dx, VIADEV_PLAYBACK + VIA_REG_OFFSET_STATUS
0001487C E808FFFFFF     <1>         call    ctrl_io_r8
                        <1>
00014881 A880           <1>       test    al, VIA_REG_STAT_ACTIVE
```

```
 977 00014883 7417                <1>         jz      short _ih0 ; 09/10/2017
 978                              <1>
 979 00014885 2407                <1>         and     al, VIA_REG_STAT_EOL + VIA_REG_STAT_FLAG + VIA_REG_STAT_STOPPED
 980 00014887 A2[45940100]        <1>       mov   [audio_flag_eol], al
 981 0001488C 740E                <1>         jz  short _ih0 ; 09/10/2017
 982                              <1>
 983                              <1>       ; 09/10/2017
 984                              <1>       ;mov  byte [audio_busy], 1
 985                              <1>
 986 0001488E 803D[44940100]01    <1>       cmp   byte [audio_play_cmd], 1
 987 00014895 7315                <1>       jnb   short _ih1 ; 10/10/2017
 988                              <1>
 989 00014897 E84A000000          <1>       call  channel_reset
 990                              <1> _ih0:
 991                              <1>       ; 09/10/2017
 992 0001489C A0[45940100]        <1>         mov     al, [audio_flag_eol]   ;; ack ;;
 993 000148A1 66BA0000            <1>         mov     dx, VIADEV_PLAYBACK + VIA_REG_OFFSET_STATUS
 994 000148A5 E8D6FEFFFF          <1>         call    ctrl_io_w8
 995 000148AA EB39                <1>       jmp   short _ih4
 996                              <1> _ih1:
 997                              <1> vt8233_tuneLoop:
 998 000148AC A0[45940100]        <1>         mov     al, [audio_flag_eol]   ;; ack ;;
 999 000148B1 66BA0000            <1>         mov     dx, VIADEV_PLAYBACK + VIA_REG_OFFSET_STATUS
1000 000148B5 E8C6FEFFFF          <1>         call    ctrl_io_w8
1001                              <1>
1002                              <1>       ; 22/07/2020
1003                              <1>       ;; 12/10/2017
1004                              <1>       ;mov  byte [audio_flag], 0 ; Reset
1005                              <1>
1006                              <1>       ; 10/10/2017
1007                              <1>       ; 09/10/2017
1008                              <1>       ;test  byte [audio_flag_eol], VIA_REG_STAT_FLAG
1009                              <1>       ;jz   short _ih2 ; EOL
1010                              <1>
1011                              <1>       ; 22/07/2020
1012                              <1>       ; 14/10/2017
1013                              <1>       ;test  byte [audio_flag_eol], VIA_REG_STAT_EOL
1014                              <1>       ;jnz  short _ih2 ; EOL
1015                              <1>       ;               ; (Half Buffer 2 has been completed
1016                              <1>       ;               ; and Half Buffer 1 will be played.)
1017                              <1>
1018                              <1>       ; FLAG
1019                              <1>       ; (Half Buffer 1 has been completed
1020                              <1>       ;  and Half Buffer 2 will be played.)
1021                              <1>
1022                              <1>       ; 14/10/2017
1023                              <1>       ;; (Continue to play.)
1024                              <1>       ;mov  al, VIA_REG_CTRL_INT
1025                              <1>             ;or   al, VIA_REG_CTRL_START
1026                              <1>             ;mov  dx, VIADEV_PLAYBACK + VIA_REG_OFFSET_CONTROL
1027                              <1>         ;call     ctrl_io_w8
1028                              <1>       ; 12/10/2017
1029                              <1>       ;mov  byte [audio_flag], 1
1030                              <1>
1031                              <1>       ; 22/07/2020
1032                              <1>       ;inc  byte [audio_flag] ; = 1
1033                              <1> _ih2:
1034                              <1>       ; 10/10/2017
1035 000148BA 8B3D[30940100]      <1>       mov   edi, [audio_dma_buff]
1036 000148C0 8B0D[34940100]      <1>       mov   ecx, [audio_dmabuff_size]
1037 000148C6 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 000148C8 F605[38940100]01    <1>       test  byte [audio_flag], 1  ; Current flag value
1047 000148CF 7402                <1>       jz    short _ih3 ; Half Buffer 1 must be filled
1048                              <1>
1049                              <1>       ; Half Buffer 2 must be filled
1050 000148D1 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 000148D3 8B35[28940100]      <1>       mov   esi, [audio_p_buffer] ; phy addr of audio buff
1056 000148D9 C1E902              <1>       shr   ecx, 2 ; half buff size / 4
1057 000148DC F3A5                <1>       rep   movsd
1058                              <1>
1059                              <1>       ; switch flag value ;
1060 000148DE 8035[38940100]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 000148E5 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 000148E6 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 000148EB B848000000          <1>         mov  eax, VIA_REG_CTRL_PAUSE + VIA_REG_CTRL_TERMINATE ; 24/06/2017
1088 000148F0 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 000148F5 B9A0000000          <1>       mov    ecx, 160 ; (200*0.25 ms) ; 29/05/2017
1095                            <1> _ch_rst_wait:
1096 000148FA E849FEFFFF          <1>       call   delay1_4ms
1097 000148FF 49                  <1>       dec    ecx
1098 00014900 75F8                <1>       jnz    short _ch_rst_wait
1099                            <1>
1100                            <1>         ; disable interrupts
1101 00014902 BA01000000          <1>         mov  edx, VIA_REG_OFFSET_CONTROL
1102 00014907 31C0                <1>         xor     eax, eax
1103 00014909 E872FEFFFF          <1>         call    ctrl_io_w8
1104                            <1>
1105                            <1>         ; clear interrupts
1106 0001490E BA00000000          <1>         mov  edx, VIA_REG_OFFSET_STATUS
1107 00014913 B803000000          <1>       mov    eax, 3
1108 00014918 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 0001491D C3                  <1>         retn
1115                            <1>
1116                            <1> vt8233_stop: ; 22/04/2017
1117 0001491E C605[44940100]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 00014925 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 00014927 D1E9                <1>       shr    ecx, 1 ; dma half buffer size
1141 00014929 89CE                <1>       mov    esi, ecx
1142                            <1>
1143 0001492B BF[4C940100]        <1>         mov     edi, audio_bdl_buff    ; get BDL address
1144 00014930 B910000000          <1>         mov     ecx, 32 / 2            ; make 32 entries in BDL
1145                            <1>
1146 00014935 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 00014937 A1[30940100]        <1>       mov    eax, [audio_dma_buff]     ; Physical address of DMA buffer
1152                            <1>
1153                            <1> s_vt8233_bdl1:
1154 0001493C AB                  <1>       stosd                    ; store dmabuffer1 address
1155                            <1>
1156 0001493D 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 0001493F 89F0                <1>       mov    eax, esi ; DMA half buffer size
1177 00014941 01C2                <1>       add    edx, eax
1178 00014943 0D00000040          <1>       or     eax, FLAG
1179                            <1>       ;or    eax, EOL
1180 00014948 AB                  <1>       stosd
1181                            <1>
1182                            <1> ; 2nd buffer:
1183                            <1>
1184 00014949 89D0                <1>         mov  eax, edx ; Physical address of the 2nd half of DMA buffer
1185 0001494B 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 0001494C 89F0             <1>       mov    eax, esi ; DMA half buffer size
1191                            <1>       ; 22/07/2020
1192                            <1>       ;or    eax, EOL
1193 0001494E 0D00000040       <1>       or     eax, FLAG
1194 00014953 AB               <1>       stosd
1195                            <1>
1196 00014954 E2E1             <1>       loop    s_vt8233_bdl0
1197                            <1>
1198                            <1>       ; 22/07/2020
1199 00014956 814FFC00000080   <1>       or     dword [edi-4], EOL
1200                            <1>
1201 0001495D 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 0001495E B82A000000        <1>       mov    eax, CODEC_EXT_AUDIO_CTRL_REG ; 2Ah
1216 00014963 E83CFEFFFF       <1>       call   codec_read
1217 00014968 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 0001496D 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 0001496F BA2A000000        <1>       mov    edx, CODEC_EXT_AUDIO_CTRL_REG
1231 00014974 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 00014979 66A1[42940100]    <1>       mov    ax, [audio_freq]
1237 0001497F BA2C000000        <1>       mov    edx, CODEC_PCM_FRONT_DACRATE_REG ; 2Ch ; PCM Front DAC Rate
1238                            <1> ;set_extd_audio_status_2:
1239 00014984 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 00014989 B8[4C940100]     <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 0001498E BA04000000        <1>       mov    edx, VIADEV_PLAYBACK + VIA_REG_OFFSET_TABLE_PTR
1255 00014993 E8FAFDFFFF       <1>         call ctrl_io_w32
1256                            <1>
1257                            <1>       ;call  codec_check_ready
1258                            <1>
1259 00014998 66BA0200          <1>       mov    dx, VIADEV_PLAYBACK + VIA_REG_OFS_PLAYBACK_VOLUME_L
1260                            <1>         ;mov eax, 2 ; 31
1261 0001499C B01F              <1>       mov    al, 31
1262 0001499E 2A05[46940100]    <1>         sub  al, [audio_master_volume_l]
1263 000149A4 E8D7FDFFFF       <1>       call   ctrl_io_w8
1264                            <1>
1265                            <1>       ;call  codec_check_ready
1266                            <1>
1267 000149A9 66BA0300          <1>         mov     dx, VIADEV_PLAYBACK + VIA_REG_OFS_PLAYBACK_VOLUME_R
1268                            <1>         ;mov ax, 2  ; 31
1269 000149AD B01F              <1>       mov    al, 31
1270 000149AF 2A05[47940100]    <1>         sub  al, [audio_master_volume_r]
1271 000149B5 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 000149BA B0FF              <1>       mov    al, 0FFh
1291 000149BC E813000000        <1>       call    set_VT8233_LastValidIndex
```

```
1292                                    <1>
1293 000149C1 C605[44940100]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 000149C8 B0A1                     <1>       mov   al, VIA_REG_CTRL_AUTOSTART + VIA_REG_CTRL_START + VIA_REG_CTRL_INT_FLAG
1305                                    <1>
1306 000149CA 66BA0100                 <1>       mov    dx, VIADEV_PLAYBACK + VIA_REG_OFFSET_CONTROL
1307 000149CE E8ADFDFFFF               <1>         call   ctrl_io_w8
1308                                    <1>       ;call  codec_check_ready
1309                                    <1>       ;retn
1310                                    <1>       ;jmp   codec_check_ready
1311 000149D3 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 000149D4 50                       <1>       push   eax ; 23/07/2020
1326                                    <1>       ;push  ecx
1327 000149D5 0FB705[42940100]         <1>       movzx  eax, word [audio_freq] ; Hertz
1328 000149DC BA00001000               <1>       mov    edx, 100000h ; 2^20 = 1048576
1329 000149E1 F7E2                     <1>       mul    edx
1330 000149E3 B980BB0000               <1>       mov    ecx, 48000
1331 000149E8 F7F1                     <1>       div    ecx
1332                                    <1>       ;and   eax, 0FFFFFh
1333                                    <1>       ;pop   ecx
1334                                    <1>       ;pop   dx
1335 000149EA 5A                       <1>       pop    edx ; 23/07/2020
1336 000149EB C1E218                   <1>       shl    edx, 24  ; STOP Index Setting: Bit 24 to 31
1337 000149EE 09D0                     <1>       or     eax, edx
1338                                    <1>       ; 19/11/2016
1339 000149F0 803D[40940100]10         <1>       cmp    byte [audio_bps], 16
1340 000149F7 7505                     <1>       jne    short sLVI_1
1341 000149F9 0D00002000               <1>       or     eax, VIA8233_REG_TYPE_16BIT
1342                                    <1> sLVI_1:
1343 000149FE 803D[41940100]02         <1>       cmp    byte [audio_stmo], 2
1344 00014A05 7505                     <1>       jne    short sLVI_2
1345 00014A07 0D00001000               <1>       or     eax, VIA8233_REG_TYPE_STEREO
1346                                    <1> sLVI_2:
1347 00014A0C BA08000000               <1>       mov     edx, VIADEV_PLAYBACK + VIA_REG_OFFSET_STOP_IDX
1348 00014A11 E87CFDFFFF               <1>         call    ctrl_io_w32
1349                                    <1>       ;call  codec_check_ready
1350                                    <1>       ;pop   edx
1351 00014A16 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 00014A17 B029                     <1>       mov    al, VIA_REG_CTRL_PAUSE+VIA_REG_CTRL_INT_FLAG+VIA_REG_CTRL_AUTOSTART
1360                                    <1>
1361 00014A19 66BA0100                 <1>       mov     dx, VIADEV_PLAYBACK + VIA_REG_OFFSET_CONTROL
1362 00014A1D E85EFDFFFF               <1>         call    ctrl_io_w8
1363                                    <1>       ;call  codec_check_ready
1364                                    <1>       ;retn
1365                                    <1>       ;jmp   codec_check_ready
1366 00014A22 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 00014A23 C605[44940100]00         <1>       mov    byte [audio_play_cmd], 0 ; stop !
1374                                    <1> vt8233_rst_0:
1375 00014A2A E8B0FCFFFF               <1>       call   reset_codec
1376 00014A2F 720A                     <1>       jc     short vt8233_rst_1 ; codec error !
1377                                    <1>       ; eax = 1
1378 00014A31 E82EFDFFFF               <1>       call   codec_io_w16 ; w32
1379 00014A36 E8ABFEFFFF               <1>       call   channel_reset
1380                                    <1> vt8233_rst_1:
1381 00014A3B 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 00014A3C 08DB                     <1>       or     bl, bl
1392 00014A3E 7520                     <1>       jnz    short vt8233_vol_1 ; temporary !
1393 00014A40 66B81F1F                 <1>       mov    ax, 1F1Fh ; 31,31
1394 00014A44 38C1                     <1>       cmp    cl, al
1395 00014A46 7718                     <1>       ja     short vt8233_vol_1 ; temporary !
1396 00014A48 38E5                     <1>       cmp    ch, ah
```

```
1397 00014A4A 7714               <1>       ja     short vt8233_vol_1 ; temporary !
1398 00014A4C 66890D[46940100]   <1>       mov    [audio_master_volume], cx
1399 00014A53 6629C8             <1>       sub    ax, cx
1400 00014A56 BA02000000         <1>       mov    edx, CODEC_MASTER_VOL_REG ; 02h ; Line Out
1401 00014A5B E870FDFFFF         <1>       call   codec_write
1402                             <1> vt8233_vol_1:
1403 00014A60 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 00014A61 66BB1002           <1>       mov     bx, 210h    ; start scanning ports
1412                             <1>                           ; 210h, 220h, .. 260h
1413                             <1> ResetDSP:
1414 00014A65 6689DA             <1>       mov     dx, bx            ; try to reset the DSP.
1415 00014A68 6683C206           <1>       add     dx, 06h
1416 00014A6C B001               <1>       mov    al, 1
1417 00014A6E EE                 <1>       out    dx, al
1418                             <1>
1419 00014A6F EC                 <1>       in     al, dx
1420 00014A70 EC                 <1>       in     al, dx
1421 00014A71 EC                 <1>       in     al, dx
1422 00014A72 EC                 <1>       in     al, dx
1423                             <1>
1424 00014A73 30C0               <1>       xor     al, al
1425 00014A75 EE                 <1>       out    dx, al
1426                             <1>
1427 00014A76 6683C208           <1>       add     dx, 08h
1428 00014A7A 66B96400           <1>       mov    cx, 100
1429                             <1> WaitID:
1430 00014A7E EC                 <1>       in     al, dx
1431 00014A7F 08C0               <1>       or      al, al
1432 00014A81 7804               <1>       js      short GetID
1433 00014A83 E2F9               <1>       loop    WaitID
1434 00014A85 EB0F               <1>       jmp     short NextPort
1435                             <1> GetID:
1436 00014A87 6683EA04           <1>       sub     dx, 04h
1437 00014A8B EC                 <1>       in     al, dx
1438 00014A8C 3CAA               <1>       cmp     al, 0AAh
1439 00014A8E 7413               <1>       je      short Found
1440 00014A90 6683C204           <1>       add     dx, 04h
1441 00014A94 E2E8               <1>       loop    WaitID
1442                             <1> NextPort:
1443 00014A96 6683C310           <1>       add     bx, 10h           ; if not response,
1444 00014A9A 6681FB6002         <1>       cmp     bx, 260h    ; try the next port.
1445 00014A9F 76C4               <1>       jbe     short ResetDSP
1446 00014AA1 F9                 <1>       stc
1447 00014AA2 C3                 <1>       retn
1448                             <1> Found:
1449 00014AA3 66891D[16940100]   <1>       mov     [audio_io_base], bx      ; SB Port Address Found!
1450                             <1> ScanIRQ:
1451                             <1> SetIrqs:
1452 00014AAA 28C0               <1>       sub    al, al ; 0
1453 00014AAC A2[0E940100]       <1>       mov    [IRQnum], al ; reset
1454 00014AB1 A2[13940100]       <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 00014AB6 66B80501           <1>       mov    ax, 105h ; IRQ 5
1463 00014ABA E8CADDFFFF         <1>       call   set_hardware_int_vector
1464 00014ABF 66B80701           <1>       mov    ax, 107h ; IRQ 7
1465 00014AC3 E8C1DDFFFF         <1>       call   set_hardware_int_vector
1466                             <1>
1467 00014AC8 668B15[16940100]   <1>       mov     dx, [audio_io_base] ; tells to the SB to
1468 00014ACF 6683C20C           <1>       add     dx, 0Ch               ; generate a IRQ!
1469                             <1> WaitSb:
1470 00014AD3 EC                 <1>       in     al, dx
1471 00014AD4 08C0               <1>       or      al, al
1472 00014AD6 78FB               <1>       js      short WaitSb
1473 00014AD8 B0F2               <1>       mov     al, 0F2h
1474 00014ADA EE                 <1>       out    dx, al
1475                             <1>
1476 00014ADB 31C9               <1>       xor     ecx, ecx    ; wait until IRQ level
1477                             <1> WaitIRQ:
1478 00014ADD A0[0E940100]       <1>       mov    al, [IRQnum]
1479 00014AE2 3C00               <1>       cmp     al, 0 ; is changed or timeout.
1480 00014AE4 7706               <1>       ja     short IrqOk
1481 00014AE6 6649               <1>       dec    cx
1482 00014AE8 75F3               <1>       jnz    short WaitIRQ
1483 00014AEA EB15               <1>       jmp    short RestoreIrqs
1484                             <1> IrqOk:
1485 00014AEC A2[13940100]       <1>       mov    [audio_intr], al ; set
1486 00014AF1 668B15[16940100]   <1>       mov     dx, [audio_io_base]
1487 00014AF8 6683C20E           <1>       add     dx, 0Eh
1488 00014AFC EC                 <1>       in     al, dx ; SB acknowledge.
1489 00014AFD B020               <1>       mov    al, 20h
1490 00014AFF 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 00014B01 66B80500           <1>       mov    ax, 5 ; IRQ 5
1500 00014B05 E87FDDFFFF         <1>       call   set_hardware_int_vector
1501 00014B0A 66B80700           <1>       mov    ax, 7 ; IRQ 7
```

```
1502 00014B0E E876DDFFFF          <1>       call   set_hardware_int_vector
1503                              <1>
1504 00014B13 31D2                <1>       xor    edx, edx
1505 00014B15 8915[18940100]      <1>       mov    [audio_dev_id], edx ; 0
1506 00014B1B 8915[1C940100]      <1>       mov    [audio_vendor], edx ; 0
1507 00014B21 8915[20940100]      <1>       mov    [audio_stats_cmd], edx ; 0
1508                              <1>
1509                              <1>       ;popad
1510                              <1>
1511 00014B27 803D[13940100]01    <1>       cmp     byte [audio_intr], 1 ; IRQ level was changed?
1512                              <1>
1513 00014B2E 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 00014B2F 8B1D[30940100]      <1>       mov    ebx, [audio_dma_buff] ; physical addr of DMA buff
1535 00014B35 89DF                <1>       mov    edi, ebx
1536 00014B37 8B0D[34940100]      <1>       mov     ecx, [audio_dmabuff_size]
1537                              <1>
1538 00014B3D 803D[40940100]10    <1>       cmp    byte [audio_bps], 16
1539 00014B44 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 00014B46 D1E9                <1>       shr    ecx, 1
1544 00014B48 49                  <1>       dec    ecx ; word count - 1
1545                              <1>       ; convert byte offset to word offset
1546 00014B49 D1EB                <1>       shr    ebx, 1
1547                              <1>
1548                              <1>       ; 16 bit DMA buffer setting (DMA channel 5)
1549 00014B4B B005                <1>       mov     al, 05h  ; set mask bit for channel 5  (4+1)
1550 00014B4D E6D4                <1>       out    0D4h, al
1551                              <1>
1552 00014B4F 30C0                <1>       xor     al, al   ; stops all DMA processes on selected channel
1553 00014B51 E6D8                <1>       out    0D8h, al ; clear selected channel register
1554                              <1>
1555 00014B53 88D8                <1>       mov     al, bl       ; byte 0 of DMA buffer offset in words (physical)
1556 00014B55 E6C4                <1>       out    0C4h, al ; DMA channel 5 port number
1557                              <1>
1558 00014B57 88F8                <1>       mov     al, bh   ; byte 1 of DMA buffer offset in words (physical)
1559 00014B59 E6C4                <1>       out    0C4h, al
1560                              <1>
1561                              <1>       ; 09/08/2017
1562 00014B5B C1EB0F              <1>       shr    ebx, 15       ; complete 16 bit shift
1563 00014B5E 80E3FE              <1>       and    bl, 0FEh ; clear bit 0 (not necessary, it will be ignored)
1564                              <1>
1565 00014B61 88D8                <1>       mov     al, bl   ; byte 2 of DMA buffer address (physical)
1566 00014B63 E68B                <1>       out    8Bh, al  ; page register port addr for channel 5 ; 13/07/2017
1567                              <1>
1568 00014B65 88C8                <1>       mov     al, cl   ; low byte of DMA count - 1
1569 00014B67 E6C6                <1>       out    0C6h, al ; count register port addr for channel 1
1570                              <1>
1571 00014B69 88E8                <1>       mov     al, ch   ; high byte of DMA count - 1
1572 00014B6B E6C6                <1>       out    0C6h, al
1573                              <1>
1574                              <1>       ; channel 5, read, autoinitialized, single mode
1575                              <1>       ;mov   al, 49h
1576 00014B6D B059                <1>       mov    al, 59h  ; 06/10/2017
1577 00014B6F E6D6                <1>       out    0D6h, al ; DMA mode register port address
1578                              <1>
1579 00014B71 B001                <1>       mov     al, 01h  ; clear mask bit for channel 1
1580 00014B73 E6D4                <1>       out    0D4h, al ; DMA mask register port address
1581                              <1>
1582 00014B75 EB28                <1>       jmp    short ClearBuffer
1583                              <1>
1584                              <1> sbInit_0:
1585 00014B77 49                  <1>       dec     ecx ; 09/08/2017
1586                              <1>
1587                              <1>       ; 8 bit DMA buffer setting (DMA channel 1)
1588 00014B78 B005                <1>       mov     al, 05h ; set mask bit for channel 1  (4+1)
1589 00014B7A E60A                <1>       out    0Ah, al ; DMA mask register
1590                              <1>
1591 00014B7C 30C0                <1>       xor     al, al  ; stops all DMA processes on selected channel
1592 00014B7E E60C                <1>       out    0Ch, al ; clear selected channel register
1593                              <1>
1594 00014B80 88D8                <1>       mov     al, bl      ; byte 0 of DMA buffer address (physical)
1595 00014B82 E602                <1>       out    02h, al ; DMA channel 1 port number
1596                              <1>
1597 00014B84 88F8                <1>       mov     al, bh  ; byte 1 of DMA buffer address (physical)
1598 00014B86 E602                <1>       out    02h, al
1599                              <1>
1600 00014B88 C1EB10              <1>       shr    ebx, 16
1601                              <1>
1602 00014B8B 88D8                <1>       mov     al, bl  ; byte 2 of DMA buffer address (physical)
1603 00014B8D E683                <1>       out    83h, al ; page register port addr for channel 1
1604                              <1>
1605 00014B8F 88C8                <1>       mov     al, cl  ; low byte of DMA count - 1
1606 00014B91 E603                <1>       out    03h, al ; count register port addr for channel 1
```

```
                                <1>
00014B93 88E8                   <1>      mov     al, ch  ; high byte of DMA count - 1
00014B95 E603                   <1>      out    03h, al
                                <1>
                                <1>      ; channel 1, read, autoinitialized, single mode
                                <1>      ;mov   al, 49h
00014B97 B059                   <1>      mov    al, 59h ; 06/10/2017
00014B99 E60B                   <1>      out    0Bh, al ; DMA mode register port address
                                <1>
00014B9B B001                   <1>      mov     al, 01h ; clear mask bit for channel 1
00014B9D E60A                   <1>      out    0Ah, al ; DMA mask register port address
                                <1>
                                <1> ClearBuffer:
                                <1>      ;;mov  edi, [audio_dma_buff]
                                <1>      ;;mov  ecx, [audio_dmabuff_size]
                                <1>      ;inc   ecx
                                <1>      ;mov     al, 80h
                                <1>      ;;cld
                                <1>      ;rep     stosb
                                <1> SetIrq:
                                <1>      ;mov   ebx, SbIrqhandler
                                <1>      ;mov   al, [audio_intr] ; IRQ number
                                <1>      ;call  set_dev_IRQ_service
                                <1>      ;; SETUP (audio) INTERRUPT CALLBACK SERVICE
                                <1>      ;mov   bl, [audio_intr] ; IRQ number
                                <1>      ;mov   bh, [audio_cb_mode]
                                <1>      ;inc   bh  ; 1 = Signal Response Byte method (fixed value)
                                <1>      ;          ; 2 = Callback service method
                                <1>      ;          ; 3 = Auto Increment S.R.B. method
                                <1>      ;mov   cl, [audio_srb]
                                <1>      ;mov   edx, [audio_cb_addr]
                                <1>      ;mov   al, [audio_user]
                                <1>      ;call  set_irq_callback_service
                                <1> ResetDsp:
00014B9F 668B15[16940100]       <1>      mov     dx, [audio_io_base]
00014BA6 6683C206               <1>      add     dx, 06h
00014BAA B001                   <1>      mov     al, 1
00014BAC EE                     <1>      out    dx, al
                                <1>
00014BAD EC                     <1>      in     al, dx
00014BAE EC                     <1>      in     al, dx
00014BAF EC                     <1>      in     al, dx
00014BB0 EC                     <1>      in     al, dx
                                <1>
00014BB1 30C0                   <1>      xor     al, al
00014BB3 EE                     <1>      out    dx, al
                                <1>
00014BB4 66B96400               <1>      mov     cx, 100
00014BB8 28E4                   <1>      sub    ah, ah ; 0
                                <1> WaitId:
00014BBA 668B15[16940100]       <1>      mov     dx, [audio_io_base]
00014BC1 6683C20E               <1>      add     dx, 0Eh
00014BC5 EC                     <1>      in     al, dx
00014BC6 08C0                   <1>      or      al, al
00014BC8 7807                   <1>      js      short sb_GetId
00014BCA E2EE                   <1>      loop    WaitId
00014BCC E9B4000000             <1>      jmp     sb_Exit
                                <1> sb_GetId:
00014BD1 668B15[16940100]       <1>      mov     dx, [audio_io_base]
00014BD8 6683C20A               <1>      add     dx, 0Ah
00014BDC EC                     <1>      in     al, dx
00014BDD 3CAA                   <1>      cmp     al, 0AAh
00014BDF 7407                   <1>      je      short SbOk
00014BE1 E2D7                   <1>      loop    WaitId
00014BE3 E99D000000             <1>      jmp    sb_Exit
                                <1> SbOk:
00014BE8 668B15[16940100]       <1>      mov     dx, [audio_io_base]
00014BEF 6683C20C               <1>      add     dx, 0Ch
                                <1>      SbOut   0D1h ; Turn on speaker
                                <2> %%Wait:
00014BF3 EC                     <2>  in al, dx
00014BF4 08C0                   <2>  or al, al
00014BF6 78FB                   <2>  js short %%Wait
00014BF8 B0D1                   <2>  mov al, %1
00014BFA EE                     <2>  out dx, al
                                <1>      SbOut   41h ; 8 bit or 16 bit transfer
                                <2> %%Wait:
00014BFB EC                     <2>  in al, dx
00014BFC 08C0                   <2>  or al, al
00014BFE 78FB                   <2>  js short %%Wait
00014C00 B041                   <2>  mov al, %1
00014C02 EE                     <2>  out dx, al
00014C03 668B1D[42940100]       <1>      mov    bx, [audio_freq] ; sampling rate (Hz)
                                <1>      SbOut  bh ; sampling rate high byte
                                <2> %%Wait:
00014C0A EC                     <2>  in al, dx
00014C0B 08C0                   <2>  or al, al
00014C0D 78FB                   <2>  js short %%Wait
00014C0F 88F8                   <2>  mov al, %1
00014C11 EE                     <2>  out dx, al
                                <1>      SbOut  bl ; sampling rate low byte
                                <2> %%Wait:
00014C12 EC                     <2>  in al, dx
00014C13 08C0                   <2>  or al, al
00014C15 78FB                   <2>  js short %%Wait
00014C17 88D8                   <2>  mov al, %1
00014C19 EE                     <2>  out dx, al
                                <1>
                                <1>      ; 22/05/2017
00014C1A E8C0000000             <1>      call   sb16_volume_initial ; 15/05/2017
                                <1>      ; 20/05/2017
                                <1>      ;call  sb16_volume
                                <1>
                                <1> StartDma:
                                <1>      ; autoinitialized mode
```

```
1688 00014C1F 803D[40940100]10    <1>       cmp    byte [audio_bps], 16 ; 16 bit samples
1689 00014C26 7411                <1>       je     short sb_play_1
1690                              <1>       ; 8 bit samples
1691 00014C28 66BBC600            <1>       mov    bx, 0C6h ; 8 bit output (0C6h)
1692 00014C2C 803D[41940100]02    <1>       cmp    byte [audio_stmo], 2 ; 1 = mono, 2 = stereo
1693 00014C33 7214                <1>       jb     short sb_play_2
1694 00014C35 B720                <1>       mov    bh, 20h      ; 8 bit stereo (20h)
1695 00014C37 EB10                <1>       jmp    short sb_play_2
1696                              <1> sb_play_1:
1697                              <1>       ; 16 bit samples
1698 00014C39 66BBB610            <1>       mov    bx, 10B6h ; 16 bit output (0B6h)
1699 00014C3D 803D[41940100]02    <1>       cmp    byte [audio_stmo], 2 ; 1 = mono, 2 = stereo
1700 00014C44 7203                <1>       jb     short sb_play_2
1701 00014C46 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 00014C49 EC                  <2>  in al, dx
1704 00014C4A 08C0                <2>  or al, al
1704 00014C4C 78FB                <2>  js short %%Wait
1704 00014C4E 88D8                <2>  mov al, %1
1704 00014C50 EE                  <2>  out dx, al
1705                              <1>       SbOut  bh ; bMode
1705                              <2> %%Wait:
1705 00014C51 EC                  <2>  in al, dx
1705 00014C52 08C0                <2>  or al, al
1705 00014C54 78FB                <2>  js short %%Wait
1705 00014C56 88F8                <2>  mov al, %1
1705 00014C58 EE                  <2>  out dx, al
1706 00014C59 8B1D[34940100]      <1>       mov    ebx, [audio_dmabuff_size]  ; 15/05/2017
1707 00014C5F D1EB                <1>       shr    ebx, 1 ; half buffer size
1708                              <1>       ; 20/10/2017
1709 00014C61 803D[40940100]10    <1>       cmp    byte [audio_bps], 16 ; 16 bit DMA
1710 00014C68 7502                <1>       jne    short sb_play_3
1711 00014C6A D1EB                <1>       shr    ebx, 1 ; byte count to word count
1712                              <1> sb_play_3:
1713 00014C6C 664B                <1>       dec    bx  ; wBlkSize is one less than the actual size
1714                              <1>       SbOut   bl
1714                              <2> %%Wait:
1714 00014C6E EC                  <2>  in al, dx
1714 00014C6F 08C0                <2>  or al, al
1714 00014C71 78FB                <2>  js short %%Wait
1714 00014C73 88D8                <2>  mov al, %1
1714 00014C75 EE                  <2>  out dx, al
1715                              <1>       SbOut   bh
1715                              <2> %%Wait:
1715 00014C76 EC                  <2>  in al, dx
1715 00014C77 08C0                <2>  or al, al
1715 00014C79 78FB                <2>  js short %%Wait
1715 00014C7B 88F8                <2>  mov al, %1
1715 00014C7D EE                  <2>  out dx, al
1716                              <1>
1717 00014C7E C605[44940100]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)
1725                              <1>       ;dec   dl
1726                              <1>       ;SbOut 31h   ; select Master Volume Register (R)
1727                              <1>       ;inc   dl
1728                              <1>       ;SbOut 0F8h  ; Max. volume value is 31 (31*8)
1729                              <1>       ;dec   dl
1730                              <1>       ;SbOut 32h   ; select Voice Volume Register (L)
1731                              <1>       ;inc   dl
1732                              <1>       ;SbOut 0F8h  ; Max. volume value is 31 (31*8)
1733                              <1>       ;dec   dl
1734                              <1>       ;SbOut 33h   ; select Voice Volume Register (R)
1735                              <1>       ;inc   dl
1736                              <1>       ;SbOut 0F8h  ; Max. volume value is 31 (31*8)
1737                              <1>       ;;
1738                              <1>       ;dec   dl
1739                              <1>       ;SbOut 44h   ; select Treble Register (L)
1740                              <1>       ;inc   dl
1741                              <1>       ;SbOut 0F0h  ; Max. Treble value is 15 (15*16)
1742                              <1>       ;dec   dl
1743                              <1>       ;SbOut 45h   ; select Treble Register (R)
1744                              <1>       ;inc   dl
1745                              <1>       ;SbOut 0F0h  ; Max. Treble value is 15 (15*16)
1746                              <1>       ;dec   dl
1747                              <1>       ;SbOut 46h   ; select Bass Register (L)
1748                              <1>       ;inc   dl
1749                              <1>       ;SbOut 0F0h  ; Max. Bass value is 15 (15*16)
1750                              <1>       ;dec   dl
1751                              <1>       ;SbOut 47h   ; select Bass Register (R)
1752                              <1>       ;inc   dl
1753                              <1>       ;SbOut 0F0h  ; Max. Bass value is 15 (15*16)
1754                              <1>
1755                              <1> sb_Exit:
1756                              <1>       ;popad
1757 00014C85 C3                  <1>       retn
1758                              <1>
1759                              <1> sb16_int_handler:
1760                              <1>       ; Interrupt Handler for Sound Blaster 16 Audio Card
1761                              <1>       ; Note: called by 'dev_IRQ_service'
1762                              <1>       ; 20/10/2017
1763                              <1>       ; 12/10/2017
1764                              <1>       ; 10/10/2017
1765                              <1>       ; 12/05/2017, 09/10/2017
1766                              <1>       ; 24/04/2017 (TRDOS 386 kernel, 'audio.s')
1767                              <1>       ; 10/03/2017 - 'PLAYWAV.PRG' ('playwav.s')
1768                              <1>
```

```
1769                                    <1>       ;push  eax ; * must be saved !
1770                                    <1>       ;push  ebx ; * must be saved !
1771                                    <1>       ;push  ecx
1772                                    <1>       ;push  edx
1773                                    <1>       ;push  esi
1774                                    <1>       ;push  edi
1775                                    <1>
1776 00014C86 668B15[16940100]         <1>       mov     dx, [audio_io_base]
1777                                    <1>       ; 20/10/2017
1778 00014C8D 80C20F                    <1>       add     dl, 0Fh ; 2xFh (DSP 16 bit intr ack)
1779 00014C90 803D[40940100]10          <1>       cmp    byte [audio_bps], 16
1780 00014C97 7402                      <1>       je     short sb_irq_16bit_ack
1781                                    <1> sb_irq_8bit_ack:
1782 00014C99 FECA                      <1>       dec    dl  ; 2xEh (DSP 8 bit intr ack)
1783                                    <1> sb_irq_16bit_ack:
1784 00014C9B EC                        <1>       in     al, dx
1785                                    <1>
1786                                    <1>       ;cmp   byte [audio_busy], 0
1787                                    <1>       ;ja    short sb_irq_h3
1788                                    <1>
1789                                    <1>       ;mov   byte [audio_busy], 1
1790                                    <1>
1791 00014C9C 803D[44940100]01          <1>       cmp    byte [audio_play_cmd], 1
1792 00014CA3 7307                      <1>       jnb    short sb_irq_h1
1793                                    <1> sb_irq_h0:
1794 00014CA5 E8A9000000                <1>       call   sb16_stop
1795 00014CAA EB2B                      <1>       jmp    short sb_irq_h3
1796                                    <1> sb_irq_h1:
1797                                    <1>       ;call  sb16_tuneloop
1798                                    <1>       ; 09/10/2017
1799                                    <1> sb16_tuneloop:
1800 00014CAC 8B3D[30940100]            <1>       mov    edi, [audio_dma_buff]
1801 00014CB2 8B0D[34940100]            <1>       mov    ecx, [audio_dmabuff_size]
1802 00014CB8 D1E9                      <1>       shr    ecx, 1 ; dma buff size / 2 = half buffer size
1803                                    <1>
1804                                    <1>       ; 22/05/2017
1805 00014CBA F605[38940100]01          <1>       test   byte [audio_flag], 1  ; Current flag value
1806 00014CC1 7402                      <1>       jz     short sb_tlp1 ; EOL (Half Buffer 1 must be filled)
1807                                    <1>       ; FLAG (Half Buffer 2 must be filled)
1808 00014CC3 01CF                      <1>       add    edi, ecx
1809                                    <1>       ; 15/05/2017
1810                                    <1> sb_tlp1:
1811 00014CC5 8B35[28940100]            <1>       mov    esi, [audio_p_buffer] ; phy addr of audio buff
1812                                    <1>       ;rep   movsb
1813 00014CCB C1E902                    <1>       shr    ecx, 2 ; half buff size / 4
1814 00014CCE F3A5                      <1>       rep    movsd
1815                                    <1>       ;retn
1816                                    <1>
1817                                    <1>       ; 10/10/2017
1818                                    <1>       ; switch flag value
1819 00014CD0 8035[38940100]01          <1>       xor    byte [audio_flag], 1
1820                                    <1>
1821                                    <1>       ; 12/10/2017
1822                                    <1>       ; [audio_flag] = 0 : Playing dma half buffer 2 (odd intr count)
1823                                    <1>                          ; Next buffer (to update) is dma half buff 1
1824                                    <1>       ;             = 1 : Playing dma half buffer 1 (even intr count)
1825                                    <1>                          ; Next buffer (to update) is dma half buff 2
1826                                    <1>
1827                                    <1> sb_irq_h3:
1828                                    <1>       ;mov   byte [audio_busy], 0
1829                                    <1>
1830                                    <1>       ;pop   edi
1831                                    <1>       ;pop   esi
1832                                    <1>       ;pop   edx
1833                                    <1>       ;pop   ecx
1834                                    <1>       ;pop   ebx ; * must be restored !
1835                                    <1>       ;pop   eax ; * must be restored !
1836                                    <1>
1837 00014CD7 C3                        <1>       retn
1838                                    <1>
1839                                    <1> sb16_volume:
1840                                    <1>       ; 22/10/2017
1841                                    <1>       ; mov [audio_master_volume_l], cl
1842                                    <1>       ; mov [audio_master_volume_h], ch
1843 00014CD8 66890D[46940100]          <1>       mov    [audio_master_volume], cx
1844                                    <1> sb16_volume_initial:
1845 00014CDF 6652                      <1>       push   dx ; DX (port address) must be saved
1846 00014CE1 668B15[16940100]          <1>       mov    dx, [audio_io_base]
1847 00014CE8 6683C204                  <1>       add    dx, 4 ; Mixer chip address port
1848 00014CEC B022                      <1>       mov    al, 22h ; master volume
1849 00014CEE EE                        <1>       out    dx, al
1850 00014CEF 6642                      <1>       inc    dx
1851 00014CF1 8A25[46940100]            <1>       mov    ah, [audio_master_volume_l]
1852 00014CF7 C0EC02                    <1>       shr    ah, 2 ; 32 -> 8 level
1853 00014CFA C0E405                    <1>       shl    ah, 5 ; bit 5 to 7
1854 00014CFD A0[47940100]              <1>       mov    al, [audio_master_volume_r]
1855 00014D02 C0E802                    <1>       shr    al, 2 ; 32 -> 8 level
1856                                    <1>       ;and   al, 0Fh
1857 00014D05 D0E0                      <1>       shl    al, 1 ; bit 1 to 3
1858 00014D07 08E0                      <1>       or     al, ah
1859 00014D09 EE                        <1>       out    dx, al
1860 00014D0A 665A                      <1>       pop    dx ; DX (port address) must be restored
1861 00014D0C C3                        <1>       retn
1862                                    <1>
1863                                    <1> sb16_pause:
1864 00014D0D 668B15[16940100]          <1>       mov    dx, [audio_io_base]
1865 00014D14 6683C20C                  <1>       add    dx, 0Ch ; Command & Data Port
1866 00014D18 803D[40940100]10          <1>       cmp    byte [audio_bps], 16 ; 16 bit samples
1867 00014D1F 7404                      <1>       je     short sb_pause_1
1868                                    <1>       ; 8 bit samples
1869 00014D21 B3D0                      <1>       mov    bl, 0D0h ; 8 bit DMA mode
1870 00014D23 EB02                      <1>       jmp    short sb_pause_2
1871                                    <1> sb_pause_1:
1872                                    <1>       ; 16 bit samples
1873 00014D25 B3D5                      <1>       mov    bl, 0D5h ; 16 bit DMA mode
```

```
1874                                    <1> sb_pause_2:
1875                                    <1>         SbOut   bl ; bCommand
1875                                    <2> %%Wait:
1875 00014D27 EC                        <2>  in al, dx
1875 00014D28 08C0                      <2>  or al, al
1875 00014D2A 78FB                      <2>  js short %%Wait
1875 00014D2C 88D8                      <2>  mov al, %1
1875 00014D2E EE                        <2>  out dx, al
1876                                    <1> sb_pause_3:
1877 00014D2F C3                        <1>         retn
1878                                    <1>
1879                                    <1> sb16_continue:
1880 00014D30 668B15[16940100]          <1>         mov    dx, [audio_io_base]
1881 00014D37 6683C20C                  <1>         add    dx, 0Ch ; Command & Data Port
1882 00014D3B 803D[40940100]10          <1>         cmp    byte [audio_bps], 16 ; 16 bit samples
1883 00014D42 7404                      <1>         je     short sb_cont_1
1884                                    <1>         ; 8 bit samples
1885 00014D44 B3D4                      <1>         mov    bl, 0D4h ; 8 bit DMA mode
1886 00014D46 EB02                      <1>         jmp    short sb_cont_2
1887                                    <1> sb_cont_1:
1888                                    <1>         ; 16 bit samples
1889 00014D48 B3D6                      <1>         mov    bl, 0D6h ; 16 bit DMA mode
1890                                    <1> sb_cont_2:
1891                                    <1>         SbOut   bl ; bCommand
1891                                    <2> %%Wait:
1891 00014D4A EC                        <2>  in al, dx
1891 00014D4B 08C0                      <2>  or al, al
1891 00014D4D 78FB                      <2>  js short %%Wait
1891 00014D4F 88D8                      <2>  mov al, %1
1891 00014D51 EE                        <2>  out dx, al
1892                                    <1> sb_cont_3:
1893 00014D52 C3                        <1>         retn
1894                                    <1>
1895                                    <1> sb16_stop:
1896                                    <1>         ; 24/04/2017
1897 00014D53 803D[44940100]00          <1>         cmp    byte [audio_play_cmd], 0
1898 00014D5A 7648                      <1>         jna    short sb16_stop_4
1899                                    <1>
1900                                    <1>         ; 22/05/2017
1901 00014D5C 668B15[16940100]          <1>         mov    dx, [audio_io_base]
1902 00014D63 6683C20C                  <1>         add    dx, 0Ch
1903                                    <1>
1904 00014D67 B3D9                      <1>         mov    bl, 0D9h ; exit auto-initialize 16 bit transfer
1905                                    <1>         ; stop  autoinitialized DMA transfer mode
1906 00014D69 803D[40940100]10          <1>         cmp    byte [audio_bps], 16 ; 16 bit samples
1907 00014D70 7402                      <1>         je     short sb16_stop_1
1908                                    <1>         ;mov   bl, 0DAh ; exit auto-initialize 8 bit transfer
1909 00014D72 FEC3                      <1>         inc    bl
1910                                    <1> sb16_stop_1:
1911                                    <1>         SbOut  bl ; exit auto-initialize transfer command
1911                                    <2> %%Wait:
1911 00014D74 EC                        <2>  in al, dx
1911 00014D75 08C0                      <2>  or al, al
1911 00014D77 78FB                      <2>  js short %%Wait
1911 00014D79 88D8                      <2>  mov al, %1
1911 00014D7B EE                        <2>  out dx, al
1912                                    <1>
1913 00014D7C 30C0                      <1>         xor     al, al ; stops all DMA processes on selected channel
1914                                    <1>
1915 00014D7E 803D[40940100]10          <1>         cmp    byte [audio_bps], 16 ; 16 bit samples
1916 00014D85 7404                      <1>         je     short sb16_stop_2
1917 00014D87 E60C                      <1>         out    0Ch, al ; clear selected channel register
1918 00014D89 EB02                      <1>         jmp    short sb16_stop_3
1919                                    <1>
1920                                    <1> sb16_stop_2:
1921 00014D8B E6D8                      <1>         out    0D8h, al ; clear selected channel register
1922                                    <1>
1923                                    <1> sb16_stop_3:
1924 00014D8D C605[44940100]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 00014D94 EC                        <2>  in al, dx
1928 00014D95 08C0                      <2>  or al, al
1928 00014D97 78FB                      <2>  js short %%Wait
1928 00014D99 B0D0                      <2>  mov al, %1
1928 00014D9B EE                        <2>  out dx, al
1929                                    <1>         SbOut   0D3h
1929                                    <2> %%Wait:
1929 00014D9C EC                        <2>  in al, dx
1929 00014D9D 08C0                      <2>  or al, al
1929 00014D9F 78FB                      <2>  js short %%Wait
1929 00014DA1 B0D3                      <2>  mov al, %1
1929 00014DA3 EE                        <2>  out dx, al
1930                                    <1> sb16_stop_4:
1931 00014DA4 C3                        <1>         retn
1932                                    <1>
1933                                    <1> sb16_reset:
1934                                    <1>         ; 24/04/2017
1935 00014DA5 668B15[16940100]          <1>         mov     dx, [audio_io_base] ; try to reset the DSP.
1936 00014DAC 6683C206                  <1>         add     dx, 06h
1937 00014DB0 B001                      <1>         mov    al, 1
1938 00014DB2 EE                        <1>         out    dx, al
1939                                    <1>
1940 00014DB3 EC                        <1>         in     al, dx
1941 00014DB4 EC                        <1>         in     al, dx
1942 00014DB5 EC                        <1>         in     al, dx
1943 00014DB6 EC                        <1>         in     al, dx
1944                                    <1>
1945 00014DB7 30C0                      <1>         xor     al, al
1946 00014DB9 EE                        <1>         out    dx, al
1947                                    <1>
1948 00014DBA 6683C208                  <1>         add     dx, 08h
```

```
1949 00014DBE 66B96400            <1>       mov    cx, 100
1950                              <1> sbrstWaitID:
1951 00014DC2 EC                  <1>       in     al, dx
1952 00014DC3 08C0                <1>       or      al, al
1953 00014DC5 7804                <1>       js      short sbrstGetID
1954 00014DC7 E2F9                <1>       loop    sbrstWaitID
1955 00014DC9 F9                  <1>       stc
1956 00014DCA C3                  <1>       retn
1957                              <1> sbrstGetID:
1958 00014DCB 6683EA04            <1>       sub     dx, 04h
1959 00014DCF EC                  <1>       in     al, dx
1960 00014DD0 3CAA                <1>       cmp     al, 0AAh
1961 00014DD2 7406                <1>       je      short sb_rst_retn
1962 00014DD4 6683C204            <1>       add     dx, 04h
1963 00014DD8 E2E8                <1>       loop    sbrstWaitID
1964                              <1> sb_rst_retn:
1965 00014DDA 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 00014DDB A1[18940100]        <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 00014DE0 B004                <1>         mov     al, PCI_CMD_REG
1999                              <1>         ;call      pciRegRead8              ; read PCI command register
2000 00014DE2 E84AF8FFFF          <1>         call pciRegRead16
2001 00014DE7 80CA05              <1>       or      dl, IO_ENA+BM_ENA              ; enable IO and bus master
2002                              <1>         ;call      pciRegWrite8
2003 00014DEA 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>
2010                              <1> ;     mov          dx, [NAMBAR]
2011                              <1> ;     add          dx, CODEC_EXT_AUDIO_CTRL_REG     ; 2Ah
2012                              <1> ;     in           ax, dx
2013                              <1> ;     or     ax, 1
2014                              <1> ;     out    dx, ax                           ; Enable variable rate audio
2015                              <1>
2016                              <1> ;       ;call    delay1_4ms
2017                              <1> ;       ;call    delay1_4ms
2018                              <1> ;       ;call    delay1_4ms
2019                              <1> ;       ;call    delay1_4ms
2020                              <1>
2021                              <1> ;     mov    ax, [audio_freq]          ; sample rate
2022                              <1>
2023                              <1> ;     mov          dx, [NAMBAR]
2024                              <1> ;     add          dx, CODEC_PCM_FRONT_DACRATE_REG  ; 2Ch
2025                              <1> ;     out    dx, ax                            ; out sample rate
2026                              <1>
2027                              <1> ;       ;call      delay1_4ms
2028                              <1> ;       ;call      delay1_4ms
2029                              <1> ;       ;call      delay1_4ms
2030                              <1> ;       ;call      delay1_4ms
2031                              <1>
2032                              <1>       ;mov   dx, [NAMBAR]              ; mixer base address
2033                              <1>         ;add dx, CODEC_RESET_REG              ; reset register
2034                              <1>         ;mov ax, 42
2035                              <1>       ;out   dx, ax                          ; reset
2036                              <1>
2037                              <1>       ;mov    dx, [NABMBAR]                  ; bus master base address
2038                              <1>         ;add dx, GLOB_STS_REG
2039                              <1>         ;mov ax, 2
2040                              <1>       ;out   dx, ax
2041                              <1>
2042 00014DEF E847F9FFFF          <1>         call    delay_100ms ; 29/05/2017
2043                              <1>
2044                              <1> init_ac97_codec:
2045                              <1>       ; 10/06/2017
2046                              <1>       ; 29/05/2017
2047                              <1>       ; 28/05/2017 - Erdogan Tan (Ref: KolibriOS, intelac97.asm)
2048                              <1>       ;
2049 00014DF4 66BA2C00            <1>       mov    dx, GLOB_CNT_REG ; 2Ch
2050 00014DF8 660315[4A940100]    <1>       add    dx, [NABMBAR]
2051 00014DFF ED                  <1>       in     eax, dx
2052                              <1>       ; ?
2053 00014E00 66BA3000            <1>       mov    dx, GLOB_STS_REG ; 30h
```

```
00014E04 660315[4A940100]    <1>       add    dx, [NABMBAR]
00014E0B ED                  <1>       in     eax, dx
                             <1>
00014E0C 83F8FF              <1>       cmp    eax, 0FFFFFFFFh ; -1
00014E0F 744B                <1>       je     short init_ac97_codec_err1
                             <1>
00014E11 A900030010          <1>       test   eax, CTRL_ST_CREADY
00014E16 7507                <1>       jnz    short _ac97_codec_ready
                             <1>
00014E18 E8EF020000          <1>       call   reset_ac97_codec
00014E1D 723E                <1>       jc     short init_ac97_codec_err2
                             <1>
                             <1> _ac97_codec_ready:
00014E1F 668B15[48940100]    <1>       mov    dx, [NAMBAR]
                             <1>       ;add   dx, 0 ; ac_reg_0 ; reset register
00014E26 66EF                <1>       out    dx, ax
                             <1>
00014E28 31C0                <1>       xor    eax, eax ; 0
00014E2A 668B15[48940100]    <1>       mov    dx, [NAMBAR]
00014E31 6683C226            <1>       add    dx, CODEC_REG_POWERDOWN
00014E35 66EF                <1>       out    dx, ax
                             <1>
                             <1>       ; 10/06/2017
                             <1>       ; 29/05/2017
                             <1>       ; wait for 1 second
00014E37 B9E8030000          <1>       mov    ecx, 1000 ; 1000*0.25ms = 1s
                             <1> _ac97_codec_rloop:
00014E3C E807F9FFFF          <1>       call   delay1_4ms
00014E41 E802F9FFFF          <1>       call   delay1_4ms
00014E46 E8FDF8FFFF          <1>       call   delay1_4ms
00014E4B E8F8F8FFFF          <1>       call   delay1_4ms
                             <1>       ;mov   dx, [NAMBAR]
                             <1>       ;add   dx, CODEC_REG_POWERDOWN
00014E50 66ED                <1>       in     ax, dx
00014E52 6683E00F            <1>       and    ax, 0Fh
00014E56 3C0F                <1>       cmp    al, 0Fh
00014E58 7404                <1>       je     short _ac97_codec_init_ok
00014E5A E2E0                <1>       loop   _ac97_codec_rloop
                             <1>
                             <1> init_ac97_codec_err1:
00014E5C F9                  <1>       stc
                             <1> init_ac97_codec_err2:
00014E5D C3                  <1>       retn
                             <1>
                             <1> _ac97_codec_init_ok:
00014E5E B002                <1>         mov     al, 2 ; force set 16-bit 2-channel PCM
00014E60 66BA2C00            <1>       mov    dx, GLOB_CNT_REG ; 2Ch
00014E64 660315[4A940100]    <1>       add    dx, [NABMBAR]
00014E6B EF                  <1>       out    dx, eax
                             <1>
                             <1>       ;call  delay1_4ms
                             <1>
                             <1>       ; 10/06/2017
00014E6C 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
00014E71 B802020000          <1>       mov     eax, 0202h
00014E76 66A3[46940100]      <1>       mov    [audio_master_volume], ax
00014E7C 66B81F1F            <1>       mov    ax, 1F1Fh ; 31, 31
                             <1>
00014E80 668B15[48940100]    <1>       mov     dx, [NAMBAR]
00014E87 6683C202            <1>       add     dx, CODEC_MASTER_VOL_REG ;02h
00014E8B 6631C0              <1>       xor    ax, ax       ; volume attenuation = 0 (max. volume)
00014E8E 66EF                <1>       out     dx, ax
                             <1>
00014E90 668B15[48940100]    <1>       mov     dx, [NAMBAR]
00014E97 6683C206            <1>       add     dx, CODEC_MASTER_MONO_VOL_REG  ;06h
                             <1>       ;xor   ax, ax
00014E9B 66EF                <1>       out     dx, ax
                             <1>
00014E9D 668B15[48940100]    <1>       mov     dx, [NAMBAR]
00014EA4 6683C20A            <1>       add     dx, CODEC_PCBEEP_VOL_REG ;0Ah
                             <1>       ;xor    ax, ax
00014EA8 66EF                <1>       out     dx, ax
                             <1>
00014EAA 668B15[48940100]    <1>       mov     dx, [NAMBAR]
00014EB1 6683C218            <1>       add     dx, CODEC_PCM_OUT_REG          ;18h
                             <1>       ;xor    ax, ax
00014EB5 66EF                <1>       out     dx, ax
                             <1>
00014EB7 66B80880            <1>       mov     ax, 8008h ; Mute
00014EBB 668B15[48940100]    <1>       mov     dx, [NAMBAR]
                             <1>       ; 22/07/2020
00014EC2 6683C20C            <1>       add    dx, CODEC_PHONE_VOL_REG          ;0Ch
                             <1>                           ; AC97_PHONE_VOL ; TAD Input (Mono)
00014EC6 66EF                <1>       out     dx, ax
                             <1>
00014EC8 66B80808            <1>         mov  ax, 0808h
00014ECC 668B15[48940100]    <1>       mov     dx, [NAMBAR]
00014ED3 6683C210            <1>         add  dx, CODEC_LINE_IN_VOL_REG ;10h ; Line Input (Stereo)
00014ED7 66EF                <1>       out     dx, ax
                             <1>
                             <1>       ;mov   ax, 0808h
00014ED9 668B15[48940100]    <1>       mov     dx, [NAMBAR]
00014EE0 6683C212            <1>         add  dx, CODEC_CD_VOL_REG ;12h ; CR Input (Stereo)
00014EE4 66EF                <1>       out     dx, ax
                             <1>
```

```
                                <1>       ;mov   ax, 0808h
00014EE6 668B15[48940100]       <1>       mov     dx, [NAMBAR]
00014EED 6683C216               <1>         add  dx, CODEC_AUX_VOL_REG ;16h ; Aux Input (Stereo)
00014EF1 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:
00014EF3 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>
00014EF4 D1E9                   <1>       shr    ecx, 1 ; dma half buffer size
00014EF6 89CE                   <1>       mov    esi, ecx
                                <1>
00014EF8 BF[4C940100]           <1>         mov     edi, audio_bdl_buff    ; get BDL address
00014EFD B910000000             <1>         mov     ecx, 32 / 2            ; make 32 entries in BDL
                                <1>
00014F02 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>
00014F04 A1[30940100]           <1>       mov    eax, [audio_dma_buff]     ; Physical address of DMA buffer
                                <1>
                                <1> s_ac97_bdl1:
00014F09 AB                     <1>       stosd                    ; store dmabuffer1 address
                                <1>
00014F0A 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>
00014F0C 89F0                   <1>       mov    eax, esi ; DMA half buffer size
00014F0E 01C2                   <1>       add    edx, eax
00014F10 D1E8                   <1>       shr    eax, 1 ; count of 16 bit samples
                                <1>       ;or    eax, IOC+BUP
00014F12 0D00000080             <1>       or     eax, IOC ; 11/06/2017
00014F17 AB                     <1>       stosd
                                <1>
                                <1> ; 2nd buffer:
                                <1>
00014F18 89D0                   <1>         mov  eax, edx ; Physical address of the 2nd half of DMA buffer
00014F1A 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> ;
00014F1B 89F0                   <1>       mov    eax, esi ; DMA half buffer size
00014F1D D1E8                   <1>       shr    eax, 1 ; count of 16 bit samples
                                <1>       ;or    eax, IOC+BUP
00014F1F 0D00000080             <1>       or     eax, IOC ; 11/06/2017
00014F24 AB                     <1>       stosd
                                <1>
00014F25 E2DD                   <1>       loop    s_ac97_bdl0
                                <1>
00014F27 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>
```

```
2264 00014F28 668B15[48940100]    <1>       mov          dx, [NAMBAR]
2265 00014F2F 6683C22A            <1>       add          dx, CODEC_EXT_AUDIO_CTRL_REG     ; 2Ah
2266 00014F33 66ED                <1>       in           ax, dx
2267 00014F35 6683C801            <1>       or     ax, 1
2268 00014F39 66EF                <1>       out    dx, ax                            ; Enable variable rate audio
2269                              <1>
2270                              <1>        ;call    delay1_4ms
2271                              <1>        ;call    delay1_4ms
2272                              <1>        ;call    delay1_4ms
2273                              <1>        ;call    delay1_4ms
2274                              <1>
2275 00014F3B 66A1[42940100]      <1>       mov    ax, [audio_freq]          ; sample rate
2276                              <1>
2277 00014F41 668B15[48940100]    <1>       mov          dx, [NAMBAR]
2278 00014F48 6683C22C            <1>       add          dx, CODEC_PCM_FRONT_DACRATE_REG  ; 2Ch
2279 00014F4C 66EF                <1>       out    dx, ax                            ; out sample rate
2280                              <1>
2281                              <1>        ;call    delay1_4ms
2282                              <1>        ;call    delay1_4ms
2283                              <1>        ;call    delay1_4ms
2284                              <1>        ;call    delay1_4ms
2285                              <1>
2286                              <1> ;
2287                              <1> ; register reset the DMA engine. This may cause a pop noise on the output
2288                              <1> ; lines when the device is reset. Prolly a better idea to mute output, then
2289                              <1> ; reset.
2290                              <1> ;
2291 00014F4E 668B15[4A940100]    <1>         mov     dx, [NABMBAR]
2292 00014F55 6683C21B            <1>         add     dx, PO_CR_REG                  ; set pointer to Cntl reg
2293 00014F59 B002                <1>         mov     al, RR                         ; set reset
2294 00014F5B EE                  <1>         out     dx, al                         ; self clearing bit
2295                              <1> ;
2296                              <1> ;     mov    edi, audio_bdl_buff
2297                              <1> ;     mov    edx, [audio_dmabuff_size]
2298                              <1> ;     shr    edx, 1
2299                              <1> ;     mov    ecx, 32/2
2300                              <1> ;ac97_set_bdl_buffer:
2301                              <1> ;     ; 1st half of DMA buffer
2302                              <1> ;     mov    eax, [audio_dma_buff]
2303                              <1> ;     push   eax
2304                              <1> ;     stosd
2305                              <1> ;     mov    eax, edx ; dma buffer size / 2
2306                              <1> ;     or     eax, IOC+BUP
2307                              <1> ;     stosd
2308                              <1> ;     pop    eax
2309                              <1> ;     ; 2nd half of DMA buffer
2310                              <1> ;     add    eax, edx
2311                              <1> ;     stosd
2312                              <1> ;     mov    eax, edx ; dma buffer size / 2
2313                              <1> ;     or     eax, IOC+BUP
2314                              <1> ;     stosd
2315                              <1> ;     loop   ac97_set_bdl_buffer
2316                              <1>
2317                              <1> ; tell the DMA engine where to find our list of Buffer Descriptors.
2318                              <1> ; this 32bit value is a flat mode memory offset (ie no segment:offset)
2319                              <1> ;
2320                              <1> ; write NABMBAR+10h with offset of buffer descriptor list
2321                              <1> ;
2322 00014F5C B8[4C940100]        <1>         mov  eax, audio_bdl_buff
2323 00014F61 668B15[4A940100]    <1>       mov    dx, [NABMBAR]
2324 00014F68 6683C210            <1>       add    dx, PO_BDBAR_REG
2325 00014F6C EF                  <1>       out    dx, eax
2326                              <1> ;
2327                              <1> ; All set. Let's play some music.
2328                              <1> ;
2329                              <1> ;
2330 00014F6D B81F000000          <1>       mov    eax, 31
2331 00014F72 E816000000          <1>       call    set_ac97_LastValidIndex
2332                              <1>
2333 00014F77 C605[44940100]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 00014F7E 668B15[4A940100]    <1>         mov     dx, [NABMBAR]
2340 00014F85 6683C21B            <1>         add     dx, PO_CR_REG          ; PCM out control register
2341 00014F89 B011                <1>         mov  al, IOCE+RPBM ; 29/05/2017
2342                              <1>         ;mov al, 1Dh ; (Ref: KolibriOS, intelac97.asm, 'play:')
2343 00014F8B 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 00014F8C 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 00014F8D 668B15[4A940100]    <1>       mov    dx, [NABMBAR]
2355 00014F94 6683C215            <1>       add    dx, PO_LVI_REG
2356 00014F98 EE                  <1>         out     dx, al
2357                              <1>       ;mov   [audio_lvi], al ; for ac97_int_handler
2358 00014F99 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 00014F9A 08DB                <1>       or     bl, bl
2367 00014F9C 7523                <1>       jnz    short ac97_vol_1 ; temporary !
2368 00014F9E 66B81F1F            <1>       mov    ax, 1F1Fh ; 31,31
```

```
2369 00014FA2 38C1                <1>       cmp    cl, al
2370 00014FA4 771B                <1>       ja     short ac97_vol_1 ; temporary !
2371 00014FA6 38E5                <1>       cmp    ch, ah
2372 00014FA8 7717                <1>       ja     short ac97_vol_1 ; temporary !
2373 00014FAA 66890D[46940100]    <1>       mov    [audio_master_volume], cx
2374 00014FB1 6629C8              <1>       sub    ax, cx
2375 00014FB4 668B15[48940100]    <1>       mov     dx, [NAMBAR]
2376 00014FBB 6683C202            <1>       add     dx, CODEC_MASTER_VOL_REG  ; 02h ; Line Out
2377 00014FBF 66EF                <1>       out     dx, ax
2378                              <1> ac97_vol_1:
2379 00014FC1 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 00014FC2 66BA3000            <1>         mov     dx, GLOB_STS_REG
2402 00014FC6 660315[4A940100]    <1>         add  dx, [NABMBAR]
2403 00014FCD ED                  <1>       in     eax, dx
2404                              <1>
2405 00014FCE 83F8FF              <1>         cmp     eax, 0FFFFFFFFh ; -1
2406 00014FD1 0F849A000000        <1>         je      _ac97_ih3 ; exit
2407                              <1>
2408 00014FD7 A940000000          <1>         test    eax, 40h ; PCM Out Interrupt
2409 00014FDC 750E                <1>         jnz     short _ac97_ih0
2410                              <1>
2411 00014FDE 85C0                <1>       test   eax, eax
2412 00014FE0 0F848B000000        <1>       jz     _ac97_ih3 ; exit
2413                              <1>
2414                              <1>       ;mov   dx, GLOB_STS_REG
2415                              <1>         ;add dx, [NABMBAR]
2416 00014FE6 EF                  <1>       out    dx, eax
2417                              <1>
2418 00014FE7 E985000000          <1>       jmp    _ac97_ih3 ; exit
2419                              <1>
2420                              <1> _ac97_ih0:
2421 00014FEC 50                  <1>       push   eax
2422                              <1>       ; 09/10/2017
2423 00014FED 803D[44940100]01    <1>       cmp    byte [audio_play_cmd], 1
2424 00014FF4 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
2430                              <1>         ;add dx, [NABMBAR]
2431                              <1>       ;out   dx, al
2432                              <1>
2433 00014FF6 66B81C00            <1>       mov    ax, 1Ch ; FIFOE(=16)+BCIS(=8)+LVBCI(=4)
2434 00014FFA 66BA1600            <1>       mov    dx, PO_SR_REG
2435 00014FFE 660315[4A940100]    <1>         add  dx, [NABMBAR]
2436 00015005 66EF                <1>       out    dx, ax
2437                              <1>
2438 00015007 66BA1400            <1>       mov    dx, PO_CIV_REG
2439 0001500B 660315[4A940100]    <1>         add  dx, [NABMBAR]
2440 00015012 EC                  <1>       in     al, dx
2441                              <1>
2442                              <1>       ;cmp   al, [audio_civ] ; [audio_flag]
2443                              <1>       ;je    short _ac97_ih2
2444                              <1>
2445 00015013 A2[45940100]        <1>       mov    [audio_civ], al
2446 00015018 FEC8                <1>       dec    al
2447                              <1>       ;inc   al ; 11/06/2017
2448 0001501A 241F                <1>       and    al, 1Fh
2449                              <1>
2450 0001501C 66BA1500            <1>         mov     dx, PO_LVI_REG
2451 00015020 660315[4A940100]    <1>         add  dx, [NABMBAR]
2452 00015027 EE                  <1>         out  dx, al
2453                              <1>
2454                              <1>       ; 12/10/2017
2455 00015028 A0[45940100]        <1>       mov    al, [audio_civ]
2456 0001502D FEC0                <1>       inc    al
2457 0001502F 2401                <1>       and    al, 1
2458 00015031 A2[38940100]        <1>       mov    [audio_flag], al
2459                              <1>       ;; [audio_flag] : 0 = Buffer 1, 1 = Buffer 2
2460                              <1>       ;
2461 00015036 58                  <1>       pop    eax
2462                              <1>       ;
2463 00015037 83E040              <1>       and     eax, 40h
2464 0001503A 668B15[4A940100]    <1>         mov  dx, [NABMBAR]
2465 00015041 6683C230            <1>       add    dx, GLOB_STS_REG
2466 00015045 EF                  <1>       out    dx, eax
2467                              <1>
2468                              <1>       ;; 13/06/2017
2469                              <1>       ;mov   al, 11h ; IOCE + RPBM
2470                              <1>       ;mov   dx, PO_CR_REG
2471                              <1>         ;add dx, [NABMBAR]
2472                              <1>       ;out   dx, al
2473                              <1>
```

```
                            <1> ac97_tuneloop:
                            <1>       ; 09/10/2017
00015046 8B3D[30940100]     <1>       mov    edi, [audio_dma_buff]
0001504C 8B0D[34940100]     <1>       mov    ecx, [audio_dmabuff_size]
00015052 D1E9               <1>       shr    ecx, 1 ; dma buff size / 2 = half buffer size
                            <1>
                            <1>       ; 12/10/2017
00015054 803D[38940100]00   <1>       cmp    byte [audio_flag], 0
0001505B 7702               <1>       ja     short _ac97_ih1  ; Playing Half Buffer 2 (Current: FLAG)
                            <1>       ; Playing Half Buffer 1 (Current: EOL)
0001505D 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>
0001505F 8B35[28940100]     <1>       mov    esi, [audio_p_buffer] ; phy addr of audio buff
00015065 C1E902             <1>       shr    ecx, 2 ; half buff size / 4
00015068 F3A5               <1>       rep    movsd
                            <1>
                            <1>       ; 10/10/2017
                            <1>       ; switch flag value
0001506A 8035[38940100]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>
00015071 C3                 <1>       retn
                            <1>
                            <1> _ac97_ih4:
                            <1>       ; 09/10/2017
00015072 E818000000         <1>       call   _ac97_stop
                            <1>       ;
00015077 58                 <1>       pop    eax
                            <1>       ;
00015078 83E040             <1>       and     eax, 40h
0001507B 668B15[4A940100]   <1>         mov  dx, [NABMBAR]
00015082 6683C230           <1>       add    dx, GLOB_STS_REG
00015086 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
00015087 C3                 <1>       retn
                            <1>
                            <1> ac97_stop:
                            <1>       ; 28/05/2017
00015088 C605[44940100]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
0001508F 30C0               <1>       xor    al, al ; 0
00015091 E813000000         <1>       call   ac97_po_cmd
                            <1>
                            <1>       ; (Ref: KolibriOS, intelac97.asm, 'stop:')
                            <1>       ; Clear FIFOE, BCIS, LVBCI (Ref: Intel ICH hub manual)
00015096 66B81C00           <1>       mov     ax, 1Ch
0001509A 668B15[4A940100]   <1>       mov     dx, [NABMBAR]
000150A1 6683C216           <1>       add     dx, PO_SR_REG
000150A5 66EF               <1>       out     dx, ax
                            <1>
                            <1>       ;retn
                            <1>
                            <1>       ; 11/06/2017
000150A7 B002               <1>       mov     al, RR
                            <1> ac97_po_cmd:
                            <1>        ;11/06/2017
                            <1>       ; 29/05/2017
000150A9 668B15[4A940100]   <1>       mov     dx, [NABMBAR]
000150B0 6683C21B           <1>         add     dx, PO_CR_REG          ; PCM out control register
000150B4 EE                 <1>       out    dx, al
000150B5 C3                 <1>       retn
                            <1>
                            <1> ac97_pause:
                            <1>       ; 11/06/2017
                            <1>       ; 29/05/2017
000150B6 B010               <1>       mov    al, IOCE
000150B8 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
```

```
2579 000150BA 31C0                <1>       xor     eax, eax
2580 000150BC 66BA0B00            <1>         mov  dx, PI_CR_REG
2581 000150C0 660315[4A940100]    <1>       add    dx, [NABMBAR]
2582 000150C7 EE                  <1>       out     dx, al
2583                              <1>
2584 000150C8 66BA1B00            <1>         mov     dx, PO_CR_REG
2585 000150CC 660315[4A940100]    <1>       add    dx, [NABMBAR]
2586 000150D3 EE                  <1>       out     dx, al
2587                              <1>
2588 000150D4 66BA2B00            <1>         mov     dx, MC_CR_REG
2589 000150D8 660315[4A940100]    <1>       add    dx, [NABMBAR]
2590 000150DF EE                  <1>       out     dx, al
2591                              <1>
2592 000150E0 B002                <1>         mov     al, RR
2593 000150E2 66BA0B00            <1>         mov     dx, PI_CR_REG
2594 000150E6 660315[4A940100]    <1>       add    dx, [NABMBAR]
2595 000150ED EE                  <1>       out     dx, al
2596                              <1>
2597 000150EE 66BA1B00            <1>         mov     dx, PO_CR_REG
2598 000150F2 660315[4A940100]    <1>       add    dx, [NABMBAR]
2599 000150F9 EE                  <1>       out     dx, al
2600                              <1>
2601 000150FA 66BA2B00            <1>         mov     dx, MC_CR_REG
2602 000150FE 660315[4A940100]    <1>       add    dx, [NABMBAR]
2603 00015105 EE                  <1>       out     dx, al
2604                              <1>
2605 00015106 C3                  <1>       retn
2606                              <1>
2607                              <1> ac97_reset:
2608                              <1>       ; 10/06/2017
2609                              <1>       ; 29/05/2017
2610                              <1>       ; 28/05/2017
2611 00015107 E8AEFFFFFF          <1>       call   reset_ac97_controller
2612                              <1>       ; 29/05/2017
2613                              <1>       ;jmp   reset_ac97_codec
2614                              <1> reset_ac97_codec:
2615                              <1>       ; 28/05/2017 - Erdogan Tan (Ref: KolibriOS, intelac97.asm)
2616 0001510C 66BA2C00            <1>       mov    dx, GLOB_CNT_REG ; 2Ch
2617 00015110 660315[4A940100]    <1>       add    dx, [NABMBAR]
2618 00015117 ED                  <1>       in     eax, dx
2619                              <1>
2620 00015118 A902000000          <1>       test   eax, 2
2621 0001511D 7407                <1>       jz     short _r_ac97codec_cold
2622                              <1>
2623 0001511F E80F000000          <1>       call   warm_ac97codec_reset
2624 00015124 7308                <1>       jnc    short _r_ac97codec_ok
2625                              <1> _r_ac97codec_cold:
2626 00015126 E83D000000          <1>         call    cold_ac97codec_reset
2627 0001512B 7301                <1>         jnc     short _r_ac97codec_ok
2628                              <1>
2629                              <1>       ; 16/04/2017
2630                              <1>         ;xor     eax, eax         ; timeout error
2631                              <1>             ;stc
2632 0001512D C3                  <1>       retn
2633                              <1>
2634                              <1> _r_ac97codec_ok:
2635 0001512E 31C0                <1>         xor     eax, eax
2636                              <1>         ;mov al, VIA_ACLINK_C00_READY ; 1
2637 00015130 FEC0                <1>         inc  al
2638 00015132 C3                  <1>       retn
2639                              <1>
2640                              <1> warm_ac97codec_reset:
2641                              <1>       ; 28/05/2017 - Erdogan Tan (Ref: KolibriOS, intelac97.asm)
2642 00015133 B806000000          <1>         mov     eax, 6
2643 00015138 66BA2C00            <1>       mov    dx, GLOB_CNT_REG ; 2Ch
2644 0001513C 660315[4A940100]    <1>       add    dx, [NABMBAR]
2645 00015143 EF                  <1>       out    dx, eax
2646                              <1>
2647 00015144 B90A000000          <1>       mov    ecx, 10      ; total 1s
2648                              <1> _warm_ac97c_rst_wait:
2649 00015149 51                  <1>       push   ecx
2650 0001514A E8ECF5FFFF          <1>       call   delay_100ms
2651 0001514F 59                  <1>       pop    ecx
2652                              <1>
2653 00015150 66BA3000            <1>       mov    dx, GLOB_STS_REG ; 30h
2654 00015154 660315[4A940100]    <1>       add    dx, [NABMBAR]
2655 0001515B ED                  <1>       in     eax, dx
2656                              <1>
2657 0001515C A900030010          <1>       test   eax, CTRL_ST_CREADY
2658 00015161 7504                <1>       jnz    short _warm_ac97c_rst_ok
2659                              <1>
2660 00015163 49                  <1>         dec     ecx
2661 00015164 75E3                <1>         jnz     short _warm_ac97c_rst_wait
2662                              <1>
2663                              <1> _warm_ac97c_rst_fail:
2664 00015166 F9                  <1>         stc
2665                              <1> _warm_ac97c_rst_ok:
2666 00015167 C3                  <1>       retn
2667                              <1>
2668                              <1> cold_ac97codec_reset:
2669                              <1>       ; 28/05/2017 - Erdogan Tan (Ref: KolibriOS, intelac97.asm)
2670 00015168 B802000000          <1>         mov     eax, 2
2671 0001516D 66BA2C00            <1>       mov    dx, GLOB_CNT_REG ; 2Ch
2672 00015171 660315[4A940100]    <1>       add    dx, [NABMBAR]
2673 00015178 EF                  <1>       out    dx, eax
2674                              <1>
2675 00015179 E8BDF5FFFF          <1>       call   delay_100ms  ; wait 100 ms
2676 0001517E E8B8F5FFFF          <1>       call   delay_100ms  ; wait 100 ms
2677 00015183 E8B3F5FFFF          <1>       call   delay_100ms  ; wait 100 ms
2678 00015188 E8AEF5FFFF          <1>       call   delay_100ms  ; wait 100 ms
2679                              <1>
2680 0001518D B910000000          <1>       mov    ecx, 16      ; total 20*100 ms = 2s
2681                              <1> _cold_ac97c_rst_wait:
2682 00015192 66BA3000            <1>       mov    dx, GLOB_STS_REG ; 30h
2683 00015196 660315[4A940100]    <1>       add    dx, [NABMBAR]
```

```
2684 0001519D ED                <1>       in     eax, dx
2685                            <1>
2686 0001519E A900030010        <1>       test   eax, CTRL_ST_CREADY
2687 000151A3 750B              <1>       jnz    short _cold_ac97c_rst_ok
2688                            <1>
2689 000151A5 51                <1>       push   ecx
2690 000151A6 E890F5FFFF        <1>       call   delay_100ms
2691 000151AB 59                <1>       pop    ecx
2692                            <1>
2693 000151AC 49                <1>         dec     ecx
2694 000151AD 75E3              <1>         jnz     short _cold_ac97c_rst_wait
2695                            <1>
2696                            <1> _cold_ac97c_rst_fail:
2697 000151AF F9                <1>         stc
2698                            <1> _cold_ac97c_rst_ok:
2699 000151B0 C3                <1>       retn
2700                            <1>
2701                            <1> sb16_current_sound_data:
2702                            <1>       ; 20/08/2017
2703                            <1>       ; 24/06/2017
2704                            <1>       ; 22/06/2017
2705                            <1>       ; get current sound (PCM out) data for graphics
2706                            <1>       ; (for Sound Blaster 16)
2707                            <1>       ; ebx = Physical address (on page boundary)
2708                            <1>       ; ecx = Byte count
2709                            <1>       ; [audio_buff_size]
2710                            <1>
2711                            <1>       ;;mov  edi, [audio_buff_size]
2712                            <1>       ;mov   edi, [audio_dmabuff_size]
2713                            <1>       ;mov   esi, [audio_dma_buff]
2714 000151B1 39CF              <1>       cmp    edi, ecx
2715 000151B3 7302              <1>       jnb    short sb16_gcd_0
2716 000151B5 89F9              <1>       mov    ecx, edi
2717                            <1> sb16_gcd_0:
2718                            <1>       ; 20/08/2017
2719 000151B7 803D[40940100]10  <1>       cmp    byte [audio_bps], 16
2720 000151BE 750F              <1>       jne    short sb16_gcd_1 ; 8 bit DMA channel
2721 000151C0 E4C6              <1>       in     al, 0C6h ; DMA channel 5 count register
2722 000151C2 88C2              <1>       mov    dl, al
2723 000151C4 E4C6              <1>       in     al, 0C6h
2724 000151C6 88C6              <1>       mov     dh, al
2725 000151C8 0FB7C2            <1>       movzx  eax, dx
2726 000151CB D1E0              <1>       shl    eax, 1 ; word count -> byte count
2727 000151CD EB4E              <1>       jmp    short sb16_gcd_2
2728                            <1> sb16_gcd_1:
2729 000151CF E403              <1>       in     al, 03h ; DMA channel 1 count register
2730 000151D1 88C2              <1>       mov    dl, al
2731 000151D3 E403              <1>       in     al, 03h
2732 000151D5 88C6              <1>       mov     dh, al
2733 000151D7 0FB7C2            <1>       movzx  eax, dx
2734 000151DA EB41              <1>       jmp    short sb16_gcd_2
2735                            <1> ;sb16_gcd_2:
2736                            <1> ;     cmp    eax, ecx
2737                            <1> ;     jnb    short sb16_gcd_3
2738                            <1> ;     ; remain count < graphics bytes
2739                            <1> ;     mov    eax, ecx ; fix remain count to data size
2740                            <1> ;sb16_gcd_3:
2741                            <1> ;     sub    edi, eax
2742                            <1> ;     jna    short sb16_gcd_4
2743                            <1> ;     add    esi, edi ; dma buffer offset
2744                            <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 000151DC 8B3D[34940100]      <1>       mov    edi, [audio_dmabuff_size]
2761 000151E2 8B35[30940100]      <1>       mov    esi, [audio_dma_buff]
2762 000151E8 803D[11940100]02    <1>       cmp    byte [audio_device], 2
2763 000151EF 72C0                <1>       jb     short sb16_current_sound_data ; = 1
2764 000151F1 D1EF                <1>       shr    edi, 1
2765 000151F3 39CF                <1>       cmp    edi, ecx
2766 000151F5 7302                <1>       jnb    short gcd_0
2767 000151F7 89F9                <1>       mov    ecx, edi
2768                              <1> gcd_0:
2769 000151F9 803D[11940100]03    <1>       cmp    byte [audio_device], 3
2770 00015200 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 00015202 BA0C000000          <1>       mov    edx, VIA_REG_OFFSET_CURR_COUNT
2790 00015207 E88FF5FFFF          <1>       call   ctrl_io_r32
2791 0001520C 89C2                <1>       mov    edx, eax ; remain count (bits 23-0),
2792                              <1>                       ; SGD index (bits 31-24)
2793 0001520E 81E200000001        <1>       and    edx, 1000000h ; SGD index (0 = 1st half)
2794 00015214 7402                <1>       jz     short vt8233_gcd_2
2795                              <1>       ; the second half of DMA buffer
2796 00015216 01FE                <1>       add    esi, edi
2797                              <1> vt8233_gcd_2:
2798 00015218 25FFFFFF00          <1>       and    eax, 0FFFFFFh ; bits 23-0
2799                              <1> ac97_gcd_2:
2800                              <1> sb16_gcd_2:
2801 0001521D 39C8                <1>       cmp    eax, ecx
2802 0001521F 7302                <1>       jnb    short vt8233_gcd_3
2803                              <1>       ; remain count < graphics bytes
2804 00015221 89C8                <1>       mov    eax, ecx ; fix remain count to data size
2805                              <1> vt8233_gcd_3:
2806 00015223 29C7                <1>       sub    edi, eax
2807 00015225 7602                <1>       jna    short vt8233_gcd_4
2808 00015227 01FE                <1>       add    esi, edi ; dma buffer offset
2809                              <1> vt8233_gcd_4:
2810 00015229 89DF                <1>       mov    edi, ebx ; buffer address (for graphics)
2811 0001522B 890D[64030300]      <1>       mov    [u.r0], ecx
2812 00015231 F3A4                <1>       rep    movsb
2813                              <1> vt8233_gcd_5:
2814 00015233 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 00015234 66BA1400              <1>       mov    dx, PO_CIV_REG ; Position In Current Buff Reg
2834 00015238 660315[4A940100]    <1>       add    dx, [NABMBAR]
2835 0001523F EC                  <1>       in     al, dx ; current index value
2836 00015240 A801                <1>       test   al, 1
2837 00015242 7402                <1>       jz     short ac97_gcd_1
2838 00015244 01FE                <1>       add    esi, edi
2839                              <1> ac97_gcd_1:
2840 00015246 31C0                <1>       xor    eax, eax
2841 00015248 66BA1800            <1>       mov    dx, PO_PICB_REG ; Position In Current Buff Reg
2842 0001524C 660315[4A940100]    <1>       add    dx, [NABMBAR]
2843 00015253 66ED                <1>       in     ax, dx ; remain dwords
2844 00015255 C1E002              <1>       shl    eax, 2 ; remain bytes ; 23/06/2017
2845 00015258 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
2850                              <1> ;ac97_gcd_2:
2851                              <1> ;     sub    edi, eax
2852                              <1> ;     jna    short ac97_gcd_3
2853                              <1> ;     add    esi, edi ; dma buffer offset
2854                              <1> ;ac97_gcd_3:
2855                              <1> ;     mov    edi, ebx ; buffer address (for graphics)
2856                              <1> ;     mov    [u.r0], ecx
2857                              <1> ;     rep    movsb
2858                              <1> ;     retn
2859                              <1>
2860                              <1> sb16_get_dma_buff_off:
2861                              <1>       ; 28/10/2017
2862                              <1>       ; 24/06/2017
2863                              <1>       ; 22/06/2017
2864                              <1>       ; get current (PCM OUT DMA buffer) pointer
2865                              <1>       ; (for Sound Blaster 16)
2866                              <1>
2867                              <1>       ;mov   ecx, [audio_dmabuff_size]
2868                              <1>       ;xor   ebx, ebx
2869                              <1>       ;shr   ecx, 1
2870                              <1> sb16_gdmabo_0:
2871                              <1>       ; 28/10/2017
2872 0001525A 803D[40940100]10    <1>       cmp    byte [audio_bps], 16
2873 00015261 750F                <1>       jne    short sb16_gdmabo_1 ; 8 bit DMA channel
2874                              <1>       ; 16 bit DMA channel
2875 00015263 E4C6                <1>       in     al, 0C6h ; DMA channel 5 count register
2876 00015265 88C2                <1>       mov    dl, al
2877 00015267 E4C6                <1>       in     al, 0C6h
2878 00015269 88C6                <1>       mov     dh, al
2879 0001526B 0FB7C2              <1>       movzx  eax, dx
2880 0001526E D1E0                <1>       shl    eax, 1 ; word count -> byte count
2881 00015270 EB3D                <1>       jmp    short sb16_gdmabo_2
2882                              <1> sb16_gdmabo_1:
2883 00015272 E403                <1>       in     al, 03h ; DMA channel 1 count register
2884 00015274 88C2                <1>       mov    dl, al
2885 00015276 E403                <1>       in     al, 03h
2886 00015278 88C6                <1>       mov     dh, al
2887 0001527A 0FB7C2              <1>       movzx  eax, dx
2888 0001527D EB30                <1>       jmp    short sb16_gdmabo_2
2889                              <1>
2890                              <1> get_dma_buffer_offset:
2891                              <1>       ; 24/06/2017
2892                              <1>       ; 22/06/2017
2893                              <1>       ; get current sound (PCM out) data for graphics
```

```
2894                                <1>       ;
2895                                <1>       ; ebx = Physical address (on page boundary)
2896                                <1>       ; ecx = Byte count
2897                                <1>       ; [audio_buff_size]
2898                                <1>
2899 0001527F 8B0D[34940100]       <1>       mov    ecx, [audio_dmabuff_size]
2900 00015285 31DB                 <1>       xor    ebx, ebx
2901                                <1> gdmabo_0:
2902 00015287 803D[11940100]02     <1>       cmp    byte [audio_device], 2
2903 0001528E 72CA                 <1>       jb     short sb16_get_dma_buff_off
2904 00015290 742A                 <1>       je     short ac97_get_dma_buff_off
2905                                <1>
2906                                <1> vt8233_get_dma_buff_off:
2907                                <1>       ; 24/06/2017
2908                                <1>       ; 22/06/2017
2909                                <1>       ; get current (PCM OUT DMA buffer) pointer
2910                                <1>       ; (for VT 8233, VT 8237R)
2911                                <1>
2912                                <1>       ;mov   ecx, [audio_dmabuff_size]
2913                                <1>       ;xor   ebx, ebx
2914 00015292 D1E9                 <1>       shr    ecx, 1
2915                                <1> vt8233_gdmabo_0:
2916 00015294 BA0C000000           <1>       mov    edx, VIA_REG_OFFSET_CURR_COUNT
2917 00015299 E8FDF4FFFF           <1>       call   ctrl_io_r32
2918 0001529E 89C2                 <1>       mov    edx, eax ; remain count (bits 23-0),
2919                                <1>                       ; SGD index (bits 31-24)
2920 000152A0 81E200000001         <1>       and    edx, 1000000h ; SGD index (0 = 1st half)
2921 000152A6 7402                 <1>       jz     short vt8233_gdmabo_1
2922                                <1>       ; the second half of DMA buffer
2923 000152A8 89CB                 <1>       mov    ebx, ecx
2924                                <1> vt8233_gdmabo_1:
2925 000152AA 25FFFFFF00           <1>       and    eax, 0FFFFFFh ; bits 23-0
2926                                <1> sb16_gdmabo_2:
2927                                <1> ac97_gdmabo_2:
2928 000152AF 29C1                 <1>       sub    ecx, eax
2929 000152B1 7602                 <1>       jna    short vt8233_gdmabo_2
2930 000152B3 01CB                 <1>       add    ebx, ecx ; dma buffer offset
2931                                <1> vt8233_gdmabo_2:
2932 000152B5 891D[64030300]       <1>       mov    [u.r0], ebx
2933 000152BB C3                   <1>       retn
2934                                <1>
2935                                <1> ac97_get_dma_buff_off:
2936                                <1>       ; 24/06/2017
2937                                <1>       ; 22/06/2017
2938                                <1>       ; get current (PCM OUT DMA buffer) pointer
2939                                <1>       ; (for AC'97, ICH)
2940                                <1>       ; ebx = Physical address (on page boundary)
2941                                <1>       ; ecx = Byte count
2942                                <1>       ; [audio_buff_size]
2943                                <1>
2944                                <1>       ;mov   ecx, [audio_dmabuff_size]
2945                                <1>       ;xor   ebx, ebx
2946 000152BC D1E9                 <1>       shr    ecx, 1
2947                                <1> ac97_gdmabo_0:
2948 000152BE 66BA1400             <1>       mov    dx, PO_CIV_REG ; Position In Current Buff Reg
2949 000152C2 660315[4A940100]     <1>       add    dx, [NABMBAR]
2950 000152C9 EC                   <1>       in     al, dx ; current index value
2951 000152CA A801                 <1>       test   al, 1
2952 000152CC 7402                 <1>       jz     short ac97_gdmabo_1
2953 000152CE 89CB                 <1>       mov    ebx, ecx
2954                                <1> ac97_gdmabo_1:
2955 000152D0 31C0                 <1>       xor    eax, eax
2956 000152D2 66BA1800             <1>       mov    dx, PO_PICB_REG ; Position In Current Buff Reg
2957 000152D6 660315[4A940100]     <1>       add    dx, [NABMBAR]
2958 000152DD 66ED                 <1>       in     ax, dx ; remain dwords
2959 000152DF EBCE                 <1>       jmp    short ac97_gdmabo_2
3425
3426 000152E1 90<rept>                  align 4
3427
3428                                    %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 000152E4 00000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
  31 000152ED 00000000000000     <1>
  32 000152F4 0000000000000002A- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah
  32 000152FD 2A2A2A2A2A2A2A     <1>
  33 00015304 2A2A2A2A2A2A2A2A2A- <1>     db  02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah
  33 0001530D 2A2A2A2A2A2A2A     <1>
  34 00015314 2A2A2A2A2A2A2A2A2A- <1>     db  02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah
  34 0001531D 2A2A2A2A2A2A2A     <1>
  35 00015324 2A2A2A2A2A2A2A2A3F- <1>     db  02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh
```

```
35 0001532D 3F3F3F3F3F3F3F      <1>
36 00015334 3F3F3F3F3F3F3F3F3F- <1>     db  03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh
36 0001533D 3F3F3F3F3F3F3F      <1>
37 00015344 000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
37 0001534D 00000000000000      <1>
38 00015354 00000000000000002A- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah
38 0001535D 2A2A2A2A2A2A2A      <1>
39 00015364 2A2A2A2A2A2A2A2A2A- <1>     db  02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah
39 0001536D 2A2A2A2A2A2A2A      <1>
40 00015374 2A2A2A2A2A2A2A2A2A- <1>     db  02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah
40 0001537D 2A2A2A2A2A2A2A      <1>
41 00015384 2A2A2A2A2A2A2A2A3F- <1>     db  02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh
41 0001538D 3F3F3F3F3F3F3F      <1>
42 00015394 3F3F3F3F3F3F3F3F3F- <1>     db  03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh
42 0001539D 3F3F3F3F3F3F3F      <1>
43                              <1> palette1: ; (63+1)*3
44 000153A4 00000000002A002A00- <1>     db  000h, 000h, 000h, 000h, 000h, 02ah, 000h, 02ah, 000h, 000h, 02ah, 02ah, 02ah, 000h, 000h, 02ah
44 000153AD 002A2A2A00002A      <1>
45 000153B4 002A2A15002A2A2A00- <1>     db  000h, 02ah, 02ah, 015h, 000h, 02ah, 02ah, 02ah, 000h, 000h, 000h, 000h, 000h, 02ah, 000h, 02ah
45 000153BD 000000002A002A      <1>
46 000153C4 00002A2A2A00002A00- <1>     db  000h, 000h, 02ah, 02ah, 02ah, 000h, 000h, 02ah, 000h, 02ah, 02ah, 015h, 000h, 02ah, 02ah, 02ah
46 000153CD 2A2A15002A2A2A      <1>
47 000153D4 15151515153F153F15- <1>     db  015h, 015h, 015h, 015h, 015h, 03fh, 015h, 03fh, 015h, 015h, 03fh, 03fh, 03fh, 015h, 015h, 03fh
47 000153DD 153F3F3F15153F      <1>
48 000153E4 153F3F3F153F3F3F15- <1>     db  015h, 03fh, 03fh, 03fh, 015h, 03fh, 03fh, 03fh, 015h, 015h, 015h, 015h, 015h, 03fh, 015h, 03fh
48 000153ED 151515153F153F      <1>
49 000153F4 15153F3F3F15153F15- <1>     db  015h, 015h, 03fh, 03fh, 03fh, 015h, 015h, 03fh, 015h, 03fh, 03fh, 03fh, 015h, 03fh, 03fh, 03fh
49 000153FD 3F3F3F153F3F3F      <1>
50 00015404 00000000002A002A00- <1>     db  000h, 000h, 000h, 000h, 000h, 02ah, 000h, 02ah, 000h, 000h, 02ah, 02ah, 02ah, 000h, 000h, 02ah
50 0001540D 002A2A2A00002A      <1>
51 00015414 002A2A15002A2A2A00- <1>     db  000h, 02ah, 02ah, 015h, 000h, 02ah, 02ah, 02ah, 000h, 000h, 000h, 000h, 000h, 02ah, 000h, 02ah
51 0001541D 000000002A002A      <1>
52 00015424 00002A2A2A00002A00- <1>     db  000h, 000h, 02ah, 02ah, 02ah, 000h, 000h, 02ah, 000h, 02ah, 02ah, 015h, 000h, 02ah, 02ah, 02ah
52 0001542D 2A2A15002A2A2A      <1>
53 00015434 15151515153F153F15- <1>     db  015h, 015h, 015h, 015h, 015h, 03fh, 015h, 03fh, 015h, 015h, 03fh, 03fh, 03fh, 015h, 015h, 03fh
53 0001543D 153F3F3F15153F      <1>
54 00015444 153F3F3F153F3F3F15- <1>     db  015h, 03fh, 03fh, 03fh, 015h, 03fh, 03fh, 03fh, 015h, 015h, 015h, 015h, 015h, 03fh, 015h, 03fh
54 0001544D 151515153F153F      <1>
55 00015454 15153F3F3F15153F15- <1>     db  015h, 015h, 03fh, 03fh, 03fh, 015h, 015h, 03fh, 015h, 03fh, 03fh, 03fh, 015h, 03fh, 03fh, 03fh
55 0001545D 3F3F3F153F3F3F      <1>
56                              <1> palette2: ; (63+1)*3
57 00015464 00000000002A002A00- <1>     db  000h, 000h, 000h, 000h, 000h, 02ah, 000h, 02ah, 000h, 000h, 02ah, 02ah, 02ah, 000h, 000h, 02ah
57 0001546D 002A2A2A00002A      <1>
58 00015474 002A2A2A002A2A2A00- <1>     db  000h, 02ah, 02ah, 02ah, 000h, 02ah, 02ah, 02ah, 000h, 000h, 015h, 000h, 000h, 03fh, 000h, 02ah
58 0001547D 001500003F002A      <1>
59 00015484 15002A3F2A00152A00- <1>     db  015h, 000h, 02ah, 03fh, 02ah, 000h, 015h, 02ah, 000h, 03fh, 02ah, 02ah, 015h, 02ah, 02ah, 03fh
59 0001548D 3F2A2A152A2A3F      <1>
60 00015494 00150000152A003F00- <1>     db  000h, 015h, 000h, 000h, 015h, 02ah, 000h, 03fh, 000h, 000h, 03fh, 02ah, 02ah, 015h, 000h, 02ah
60 0001549D 003F2A2A15002A      <1>
61 000154A4 152A2A3F002A3F2A00- <1>     db  015h, 02ah, 02ah, 03fh, 000h, 02ah, 03fh, 02ah, 000h, 015h, 015h, 000h, 015h, 03fh, 000h, 03fh
61 000154AD 151500153F003F      <1>
62 000154B4 15003F3F2A15152A15- <1>     db  015h, 000h, 03fh, 03fh, 02ah, 015h, 015h, 02ah, 015h, 03fh, 02ah, 03fh, 015h, 02ah, 03fh, 03fh
62 000154BD 3F2A3F152A3F3F      <1>
63 000154C4 15000015002A152A00- <1>     db  015h, 000h, 000h, 015h, 000h, 02ah, 015h, 02ah, 000h, 015h, 02ah, 02ah, 03fh, 000h, 000h, 03fh
63 000154CD 152A2A3F00003F      <1>
64 000154D4 002A3F2A003F2A2A15- <1>     db  000h, 02ah, 03fh, 02ah, 000h, 03fh, 02ah, 02ah, 015h, 000h, 015h, 015h, 000h, 03fh, 015h, 02ah
64 000154DD 001515003F152A      <1>
65 000154E4 15152A3F3F00153F00- <1>     db  015h, 015h, 02ah, 03fh, 03fh, 000h, 015h, 03fh, 000h, 03fh, 03fh, 02ah, 015h, 03fh, 02ah, 03fh
65 000154ED 3F3F2A153F2A3F      <1>
66 000154F4 15150015152A153F00- <1>     db  015h, 015h, 000h, 015h, 015h, 02ah, 015h, 03fh, 000h, 015h, 03fh, 02ah, 03fh, 015h, 000h, 03fh
66 000154FD 153F2A3F15003F      <1>
67 00015504 152A3F3F003F3F2A15- <1>     db  015h, 02ah, 03fh, 03fh, 000h, 03fh, 03fh, 02ah, 015h, 015h, 015h, 015h, 015h, 03fh, 015h, 03fh
67 0001550D 151515153F153F      <1>
68 00015514 15153F3F3F15153F15- <1>     db  015h, 015h, 03fh, 03fh, 03fh, 015h, 015h, 03fh, 015h, 03fh, 03fh, 03fh, 015h, 03fh, 03fh, 03fh
68 0001551D 3F3F3F153F3F3F      <1>
69                              <1> palette3: ; 256*3
70 00015524 00000000002A002A00- <1>     db  000h, 000h, 000h, 000h, 000h, 02ah, 000h, 02ah, 000h, 000h, 02ah, 02ah, 02ah, 000h, 000h, 02ah
70 0001552D 002A2A2A00002A      <1>
71 00015534 002A2A15002A2A2A15- <1>     db  000h, 02ah, 02ah, 015h, 000h, 02ah, 02ah, 02ah, 015h, 015h, 015h, 015h, 015h, 03fh, 015h, 03fh
71 0001553D 151515153F153F      <1>
72 00015544 15153F3F3F15153F15- <1>     db  015h, 015h, 03fh, 03fh, 03fh, 015h, 015h, 03fh, 015h, 03fh, 03fh, 03fh, 015h, 03fh, 03fh, 03fh
72 0001554D 3F3F3F153F3F3F      <1>
73 00015554 00000005050508080- <1>     db  000h, 000h, 000h, 005h, 005h, 005h, 008h, 008h, 008h, 00bh, 00bh, 00bh, 00eh, 00eh, 00eh, 011h
73 0001555D 0B0B0B0E0E0E11      <1>
74 00015564 11111414141818181C- <1>     db  011h, 011h, 014h, 014h, 014h, 018h, 018h, 018h, 01ch, 01ch, 01ch, 020h, 020h, 020h, 024h, 024h
74 0001556D 1C1C2020202424      <1>
75 00015574 242828282D2D2D3232- <1>     db  024h, 028h, 028h, 028h, 02dh, 02dh, 02dh, 032h, 032h, 032h, 038h, 038h, 038h, 03fh, 03fh, 03fh
75 0001557D 323838383F3F3F      <1>
76 00015584 00003F10003F1F003F- <1>     db  000h, 000h, 03fh, 010h, 000h, 03fh, 01fh, 000h, 03fh, 02fh, 000h, 03fh, 03fh, 000h, 03fh, 03fh
76 0001558D 2F003F3F003F3F      <1>
77 00015594 002F3F001F3F00103F- <1>     db  000h, 02fh, 03fh, 000h, 01fh, 03fh, 000h, 010h, 03fh, 000h, 000h, 03fh, 010h, 000h, 03fh, 01fh
77 0001559D 00003F10003F1F      <1>
78 000155A4 003F2F003F3F002F3F- <1>     db  000h, 03fh, 02fh, 000h, 03fh, 03fh, 000h, 02fh, 03fh, 000h, 01fh, 03fh, 000h, 010h, 03fh, 000h
78 000155AD 001F3F00103F00      <1>
79 000155B4 003F00003F10003F1F- <1>     db  000h, 03fh, 000h, 000h, 03fh, 010h, 000h, 03fh, 01fh, 000h, 03fh, 02fh, 000h, 03fh, 03fh, 000h
79 000155BD 003F2F003F3F00      <1>
80 000155C4 2F3F001F3F00103F1F- <1>     db  02fh, 03fh, 000h, 01fh, 03fh, 000h, 010h, 03fh, 01fh, 01fh, 03fh, 027h, 01fh, 03fh, 02fh, 01fh
80 000155CD 1F3F271F3F2F1F      <1>
81 000155D4 3F371F3F3F1F3F3F1F- <1>     db  03fh, 037h, 01fh, 03fh, 03fh, 01fh, 03fh, 03fh, 01fh, 037h, 03fh, 01fh, 02fh, 03fh, 01fh, 027h
81 000155DD 373F1F2F3F1F27      <1>
82 000155E4 3F1F1F3F271F3F2F1F- <1>     db  03fh, 01fh, 01fh, 03fh, 027h, 01fh, 03fh, 02fh, 01fh, 03fh, 037h, 01fh, 03fh, 03fh, 01fh, 037h
82 000155ED 3F371F3F3F1F37      <1>
83 000155F4 3F1F2F3F1F273F1F1F- <1>     db  03fh, 01fh, 02fh, 03fh, 01fh, 027h, 03fh, 01fh, 01fh, 03fh, 01fh, 01fh, 03fh, 027h, 01fh, 03fh
83 000155FD 3F1F1F3F271F3F      <1>
84 00015604 2F1F3F371F3F3F1F37- <1>     db  02fh, 01fh, 03fh, 037h, 01fh, 03fh, 03fh, 01fh, 037h, 03fh, 01fh, 02fh, 03fh, 01fh, 027h, 03fh
84 0001560D 3F1F2F3F1F273F      <1>
85 00015614 2D2D3F312D3F362D3F- <1>     db  02dh, 02dh, 03fh, 031h, 02dh, 03fh, 036h, 02dh, 03fh, 03ah, 02dh, 03fh, 03fh, 02dh, 03fh, 03fh
85 0001561D 3A2D3F3F2D3F3F      <1>
86 00015624 2D3A3F2D363F2D313F- <1>     db  02dh, 03ah, 03fh, 02dh, 036h, 03fh, 02dh, 031h, 03fh, 02dh, 02dh, 03fh, 031h, 02dh, 03fh, 036h
86 0001562D 2D2D3F312D3F36      <1>
87 00015634 2D3F3A2D3F3F2D3A3F- <1>     db  02dh, 03fh, 03ah, 02dh, 03fh, 03fh, 02dh, 03ah, 03fh, 02dh, 036h, 03fh, 02dh, 031h, 03fh, 02dh
87 0001563D 2D363F2D313F2D      <1>
88 00015644 2D3F2D2D3F312D3F36- <1>     db  02dh, 03fh, 02dh, 02dh, 03fh, 031h, 02dh, 03fh, 036h, 02dh, 03fh, 03ah, 02dh, 03fh, 03fh, 02dh
88 0001564D 2D3F3A2D3F3F2D      <1>
89 00015654 3A3F2D363F2D313F00- <1>     db  03ah, 03fh, 02dh, 036h, 03fh, 02dh, 031h, 03fh, 000h, 000h, 01ch, 007h, 000h, 01ch, 00eh, 000h
89 0001565D 001C07001C0E00      <1>
90 00015664 1C15001C1C001C1C00- <1>     db  01ch, 015h, 000h, 01ch, 01ch, 000h, 01ch, 01ch, 000h, 015h, 01ch, 000h, 00eh, 01ch, 000h, 007h
90 0001566D 151C000E1C0007      <1>
91 00015674 1C00001C07001C0E00- <1>     db  01ch, 000h, 000h, 01ch, 007h, 000h, 01ch, 00eh, 000h, 01ch, 015h, 000h, 01ch, 01ch, 000h, 015h
91 0001567D 1C15001C1C0015      <1>
92 00015684 1C000E1C00071C0000- <1>     db  01ch, 000h, 00eh, 01ch, 000h, 007h, 01ch, 000h, 000h, 01ch, 000h, 000h, 01ch, 007h, 000h, 01ch
92 0001568D 1C00001C07001C      <1>
93 00015694 0E001C15001C1C0015- <1>     db  00eh, 000h, 01ch, 015h, 000h, 01ch, 01ch, 000h, 015h, 01ch, 000h, 00eh, 01ch, 000h, 007h, 01ch
93 0001569D 1C000E1C00071C      <1>
94 000156A4 0E0E1C110E1C150E1C- <1>     db  00eh, 00eh, 01ch, 011h, 00eh, 01ch, 015h, 00eh, 01ch, 018h, 00eh, 01ch, 01ch, 00eh, 01ch, 01ch
94 000156AD 180E1C1C0E1C1C      <1>
95 000156B4 0E181C0E151C0E111C- <1>     db  00eh, 018h, 01ch, 00eh, 015h, 01ch, 00eh, 011h, 01ch, 00eh, 00eh, 01ch, 011h, 00eh, 01ch, 015h
95 000156BD 0E0E1C110E1C15      <1>
```

```
 96 000156C4 0E1C180E1C1C0E181C- <1>     db  00eh, 01ch, 018h, 00eh, 01ch, 01ch, 00eh, 018h, 01ch, 00eh, 015h, 01ch, 00eh, 011h, 01ch, 00eh
 96 000156CD 0E151C0E111C0E      <1>
 97 000156D4 0E1C0E0E1C110E1C15- <1>     db  00eh, 01ch, 00eh, 00eh, 01ch, 011h, 00eh, 01ch, 015h, 00eh, 01ch, 018h, 00eh, 01ch, 01ch, 00eh
 97 000156DD 0E1C180E1C1C0E      <1>
 98 000156E4 181C0E151C0E111C14- <1>     db  018h, 01ch, 00eh, 015h, 01ch, 00eh, 011h, 01ch, 014h, 014h, 01ch, 016h, 014h, 01ch, 018h, 014h
 98 000156ED 141C16141C1814      <1>
 99 000156F4 1C1A141C1C141C1C14- <1>     db  01ch, 01ah, 014h, 01ch, 01ch, 014h, 01ch, 01ch, 014h, 01ah, 01ch, 014h, 018h, 01ch, 014h, 016h
 99 000156FD 1A1C14181C1416      <1>
100 00015704 1C14141C16141C1814- <1>     db  01ch, 014h, 014h, 01ch, 016h, 014h, 01ch, 018h, 014h, 01ch, 01ah, 014h, 01ch, 01ch, 014h, 01ah
100 0001570D 1C1A141C1C141A      <1>
101 00015714 1C14181C14161C1414- <1>     db  01ch, 014h, 018h, 01ch, 014h, 016h, 01ch, 014h, 014h, 01ch, 014h, 014h, 01ch, 016h, 014h, 01ch
101 0001571D 1C14141C16141C      <1>
102 00015724 18141C1A141C1C141A- <1>     db  018h, 014h, 01ch, 01ah, 014h, 01ch, 01ch, 014h, 01ah, 01ch, 014h, 018h, 01ch, 014h, 016h, 01ch
102 0001572D 1C14181C14161C      <1>
103 00015734 000010040010080010- <1>     db  000h, 000h, 010h, 004h, 000h, 010h, 008h, 000h, 010h, 00ch, 000h, 010h, 010h, 000h, 010h, 010h
103 0001573D 0C001010001010      <1>
104 00015744 000C10000810000410- <1>     db  000h, 00ch, 010h, 000h, 008h, 010h, 000h, 004h, 010h, 000h, 000h, 010h, 004h, 000h, 010h, 008h
104 0001574D 00001004001008      <1>
105 00015754 00100C001010000C10- <1>     db  000h, 010h, 00ch, 000h, 010h, 010h, 000h, 00ch, 010h, 000h, 008h, 010h, 000h, 004h, 010h, 000h
105 0001575D 00081000041000      <1>
106 00015764 001000001004001008- <1>     db  000h, 010h, 000h, 000h, 010h, 004h, 000h, 010h, 008h, 000h, 010h, 00ch, 000h, 010h, 010h, 000h
106 0001576D 00100C00101000      <1>
107 00015774 0C1000081000041008- <1>     db  00ch, 010h, 000h, 008h, 010h, 000h, 004h, 010h, 008h, 008h, 010h, 00ah, 008h, 010h, 00ch, 008h
107 0001577D 08100A08100C08      <1>
108 00015784 100E08101008101008- <1>     db  010h, 00eh, 008h, 010h, 010h, 008h, 010h, 010h, 008h, 00eh, 010h, 008h, 00ch, 010h, 008h, 00ah
108 0001578D 0E10080C10080A      <1>
109 00015794 100808100A08100C08- <1>     db  010h, 008h, 008h, 010h, 00ah, 008h, 010h, 00ch, 008h, 010h, 00eh, 008h, 010h, 010h, 008h, 00eh
109 0001579D 100E081010080E      <1>
110 000157A4 10080C10080A100808- <1>     db  010h, 008h, 00ch, 010h, 008h, 00ah, 010h, 008h, 008h, 010h, 008h, 008h, 010h, 00ah, 008h, 010h
110 000157AD 100808100A0810      <1>
111 000157B4 0C08100E081010080E- <1>     db  00ch, 008h, 010h, 00eh, 008h, 010h, 010h, 008h, 00eh, 010h, 008h, 00ch, 010h, 008h, 00ah, 010h
111 000157BD 10080C10080A10      <1>
112 000157C4 0B0B100C0B100D0B10- <1>     db  00bh, 00bh, 010h, 00ch, 00bh, 010h, 00dh, 00bh, 010h, 00fh, 00bh, 010h, 010h, 00bh, 010h, 010h
112 000157CD 0F0B10100B1010      <1>
113 000157D4 0B0F100B0D100B0C10- <1>     db  00bh, 00fh, 010h, 00bh, 00dh, 010h, 00bh, 00ch, 010h, 00bh, 00bh, 010h, 00ch, 00bh, 010h, 00dh
113 000157DD 0B0B100C0B100D      <1>
114 000157E4 0B100F0B10100B0F10- <1>     db  00bh, 010h, 00fh, 00bh, 010h, 010h, 00bh, 00fh, 010h, 00bh, 00dh, 010h, 00bh, 00ch, 010h, 00bh
114 000157ED 0B0D100B0C100B      <1>
115 000157F4 0B100B0B100C0B100D- <1>     db  00bh, 010h, 00bh, 00bh, 010h, 00ch, 00bh, 010h, 00dh, 00bh, 010h, 00fh, 00bh, 010h, 010h, 00bh
115 000157FD 0B100F0B10100B      <1>
116 00015804 0F100B0D100B0C1000- <1>     db  00fh, 010h, 00bh, 00dh, 010h, 00bh, 00ch, 010h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
116 0001580D 00000000000000      <1>
117 00015814 000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
117 0001581D 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 00015824 000000000000007E- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 07eh, 081h, 0a5h, 081h, 0bdh, 099h, 081h, 07eh
125 0001582D 81A581BD99817E      <1>
126 00015834 7EFFDBFFC3E7FF7E6C- <1>     db  07eh, 0ffh, 0dbh, 0ffh, 0c3h, 0e7h, 0ffh, 07eh, 06ch, 0feh, 0feh, 0feh, 07ch, 038h, 010h, 000h
126 0001583D FEFEFE7C381000      <1>
127 00015844 10387CFE7C38100038- <1>     db  010h, 038h, 07ch, 0feh, 07ch, 038h, 010h, 000h, 038h, 07ch, 038h, 0feh, 0feh, 07ch, 038h, 07ch
127 0001584D 7C38FEFE7C387C      <1>
128 00015854 1010387CFE7C387C00- <1>     db  010h, 010h, 038h, 07ch, 0feh, 07ch, 038h, 07ch, 000h, 000h, 018h, 03ch, 03ch, 018h, 000h, 000h
128 0001585D 00183C3C180000      <1>
129 00015864 FFFFE7C3C3E7FFFF00- <1>     db  0ffh, 0ffh, 0e7h, 0c3h, 0c3h, 0e7h, 0ffh, 0ffh, 000h, 03ch, 066h, 042h, 042h, 066h, 03ch, 000h
129 0001586D 3C664242663C00      <1>
130 00015874 FFC399BDBD99C3FF0F- <1>     db  0ffh, 0c3h, 099h, 0bdh, 0bdh, 099h, 0c3h, 0ffh, 00fh, 007h, 00fh, 07dh, 0cch, 0cch, 0cch, 078h
130 0001587D 070F7DCCCCCC78      <1>
131 00015884 3C6666663C187E183F- <1>     db  03ch, 066h, 066h, 066h, 03ch, 018h, 07eh, 018h, 03fh, 033h, 03fh, 030h, 030h, 070h, 0f0h, 0e0h
131 0001588D 333F303070F0E0      <1>
132 00015894 7F637F636367E6C099- <1>     db  07fh, 063h, 07fh, 063h, 063h, 067h, 0e6h, 0c0h, 099h, 05ah, 03ch, 0e7h, 0e7h, 03ch, 05ah, 099h
132 0001589D 5A3CE7E73C5A99      <1>
133 000158A4 80E0F8FEF8E0800002- <1>     db  080h, 0e0h, 0f8h, 0feh, 0f8h, 0e0h, 080h, 000h, 002h, 00eh, 03eh, 0feh, 03eh, 00eh, 002h, 000h
133 000158AD 0E3EFE3E0E0200      <1>
134 000158B4 183C7E18187E3C1866- <1>     db  018h, 03ch, 07eh, 018h, 018h, 07eh, 03ch, 018h, 066h, 066h, 066h, 066h, 066h, 000h, 066h, 000h
134 000158BD 66666666006600      <1>
135 000158C4 7FDBDB7B1B1B1B003E- <1>     db  07fh, 0dbh, 0dbh, 07bh, 01bh, 01bh, 01bh, 000h, 03eh, 063h, 038h, 06ch, 06ch, 038h, 0cch, 078h
135 000158CD 63386C6C38CC78      <1>
136 000158D4 000000007E7E7E0018- <1>     db  000h, 000h, 000h, 000h, 07eh, 07eh, 07eh, 000h, 018h, 03ch, 07eh, 018h, 07eh, 03ch, 018h, 0ffh
136 000158DD 3C7E187E3C18FF      <1>
137 000158E4 183C7E1818180018- <1>     db  018h, 03ch, 07eh, 018h, 018h, 018h, 018h, 000h, 018h, 018h, 018h, 018h, 07eh, 03ch, 018h, 000h
137 000158ED 1818187E3C1800      <1>
138 000158F4 00180CFE0C18000000- <1>     db  000h, 018h, 00ch, 0feh, 00ch, 018h, 000h, 000h, 000h, 030h, 060h, 0feh, 060h, 030h, 000h, 000h
138 000158FD 3060FE60300000      <1>
139 00015904 0000C0C0C0FE000000- <1>     db  000h, 000h, 0c0h, 0c0h, 0c0h, 0feh, 000h, 000h, 000h, 024h, 066h, 0ffh, 066h, 024h, 000h, 000h
139 0001590D 2466FF66240000      <1>
140 00015914 00183C7EFFFF000000- <1>     db  000h, 018h, 03ch, 07eh, 0ffh, 0ffh, 000h, 000h, 000h, 0ffh, 0ffh, 07eh, 03ch, 018h, 000h, 000h
140 0001591D FFFF7E3C180000      <1>
141 00015924 000000000000000030- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 030h, 078h, 078h, 030h, 030h, 000h, 030h, 000h
141 0001592D 78783030003000      <1>
142 00015934 6C6C6C000000000006C- <1>     db  06ch, 06ch, 06ch, 000h, 000h, 000h, 000h, 000h, 06ch, 06ch, 0feh, 06ch, 0feh, 06ch, 06ch, 000h
142 0001593D 6CFE6CFE6C6C00      <1>
143 00015944 307CC0780CF8300000- <1>     db  030h, 07ch, 0c0h, 078h, 00ch, 0f8h, 030h, 000h, 000h, 0c6h, 0cch, 018h, 030h, 066h, 0c6h, 000h
143 0001594D C6CC183066C600      <1>
144 00015954 386C3876DCCC760060- <1>     db  038h, 06ch, 038h, 076h, 0dch, 0cch, 076h, 000h, 060h, 060h, 0c0h, 000h, 000h, 000h, 000h, 000h
144 0001595D 60C00000000000      <1>
145 00015964 183060606030180060- <1>     db  018h, 030h, 060h, 060h, 060h, 030h, 018h, 000h, 060h, 030h, 018h, 018h, 018h, 030h, 060h, 000h
145 0001596D 30181818306000      <1>
146 00015974 00663CFF3C66000000- <1>     db  000h, 066h, 03ch, 0ffh, 03ch, 066h, 000h, 000h, 000h, 030h, 030h, 0fch, 030h, 030h, 000h, 000h
146 0001597D 3030FC30300000      <1>
147 00015984 000000000030306000- <1>     db  000h, 000h, 000h, 000h, 000h, 030h, 030h, 060h, 000h, 000h, 000h, 0fch, 000h, 000h, 000h, 000h
147 0001598D 0000FC00000000      <1>
148 00015994 000000000030300006- <1>     db  000h, 000h, 000h, 000h, 000h, 030h, 030h, 000h, 006h, 00ch, 018h, 030h, 060h, 0c0h, 080h, 000h
148 0001599D 0C183060C08000      <1>
149 000159A4 7CC6CEDEF6E67C0030- <1>     db  07ch, 0c6h, 0ceh, 0deh, 0f6h, 0e6h, 07ch, 000h, 030h, 070h, 030h, 030h, 030h, 030h, 0fch, 000h
149 000159AD 7030303030FC00      <1>
150 000159B4 78CC0C3860CCFC0078- <1>     db  078h, 0cch, 00ch, 038h, 060h, 0cch, 0fch, 000h, 078h, 0cch, 00ch, 038h, 00ch, 0cch, 078h, 000h
150 000159BD CC0C380CCC7800      <1>
151 000159C4 1C3C6CCCFE0C1E00FC- <1>     db  01ch, 03ch, 06ch, 0cch, 0feh, 00ch, 01eh, 000h, 0fch, 0c0h, 0f8h, 00ch, 00ch, 0cch, 078h, 000h
151 000159CD C0F80C0CCC7800      <1>
152 000159D4 3860C0F8CCCC7800FC- <1>     db  038h, 060h, 0c0h, 0f8h, 0cch, 0cch, 078h, 000h, 0fch, 0cch, 00ch, 018h, 030h, 030h, 030h, 000h
152 000159DD CC0C1830303000      <1>
153 000159E4 78CCCC78CCCC780078- <1>     db  078h, 0cch, 0cch, 078h, 0cch, 0cch, 078h, 000h, 078h, 0cch, 0cch, 07ch, 00ch, 018h, 070h, 000h
153 000159ED CCCC7C0C187000      <1>
154 000159F4 003030000030300000- <1>     db  000h, 030h, 030h, 000h, 000h, 030h, 030h, 000h, 000h, 030h, 030h, 000h, 000h, 030h, 030h, 060h
154 000159FD 30300000303060      <1>
155 00015A04 183060C06030180000- <1>     db  018h, 030h, 060h, 0c0h, 060h, 030h, 018h, 000h, 000h, 000h, 0fch, 000h, 000h, 0fch, 000h, 000h
155 00015A0D 00FC0000FC0000      <1>
156 00015A14 6030180C1830600078- <1>     db  060h, 030h, 018h, 00ch, 018h, 030h, 060h, 000h, 078h, 0cch, 00ch, 018h, 030h, 000h, 030h, 000h
156 00015A1D CC0C1830003000      <1>
157 00015A24 7CC6DEDEDEC0780030- <1>     db  07ch, 0c6h, 0deh, 0deh, 0deh, 0c0h, 078h, 000h, 030h, 078h, 0cch, 0cch, 0fch, 0cch, 0cch, 000h
157 00015A2D 78CCCCFCCCCC00      <1>
158 00015A34 FC66667C6666FC003C- <1>     db  0fch, 066h, 066h, 07ch, 066h, 066h, 0fch, 000h, 03ch, 066h, 0c0h, 0c0h, 0c0h, 066h, 03ch, 000h
```

```
158 00015A3D 66C0C0C0663C00      <1>
159 00015A44 F86C6666666CF800FE- <1>     db  0f8h, 06ch, 066h, 066h, 066h, 06ch, 0f8h, 000h, 0feh, 062h, 068h, 078h, 068h, 062h, 0feh, 000h
159 00015A4D 6268786862FE00      <1>
160 00015A54 FE6268786860F0003C- <1>     db  0feh, 062h, 068h, 078h, 068h, 060h, 0f0h, 000h, 03ch, 066h, 0c0h, 0c0h, 0ceh, 066h, 03eh, 000h
160 00015A5D 66C0C0CE663E00      <1>
161 00015A64 CCCCCCFCCCCCCC0078- <1>     db  0cch, 0cch, 0cch, 0fch, 0cch, 0cch, 0cch, 000h, 078h, 030h, 030h, 030h, 030h, 030h, 078h, 000h
161 00015A6D 30303030307800      <1>
162 00015A74 1E0C0C0CCCCC7800E6- <1>     db  01eh, 00ch, 00ch, 00ch, 0cch, 0cch, 078h, 000h, 0e6h, 066h, 06ch, 078h, 06ch, 066h, 0e6h, 000h
162 00015A7D 666C786C66E600      <1>
163 00015A84 F06060606266FE00C6- <1>     db  0f0h, 060h, 060h, 060h, 062h, 066h, 0feh, 000h, 0c6h, 0eeh, 0feh, 0feh, 0d6h, 0c6h, 0c6h, 000h
163 00015A8D EEFEFED6C6C600      <1>
164 00015A94 C6E6F6DECEC6C60038- <1>     db  0c6h, 0e6h, 0f6h, 0deh, 0ceh, 0c6h, 0c6h, 000h, 038h, 06ch, 0c6h, 0c6h, 0c6h, 06ch, 038h, 000h
164 00015A9D 6CC6C6C66C3800      <1>
165 00015AA4 FC66667C6060F00078- <1>     db  0fch, 066h, 066h, 07ch, 060h, 060h, 0f0h, 000h, 078h, 0cch, 0cch, 0cch, 0dch, 078h, 01ch, 000h
165 00015AAD CCCCCCDC781C00      <1>
166 00015AB4 FC66667C6C66E60078- <1>     db  0fch, 066h, 066h, 07ch, 06ch, 066h, 0e6h, 000h, 078h, 0cch, 0e0h, 070h, 01ch, 0cch, 078h, 000h
166 00015ABD CCE0701CCC7800      <1>
167 00015AC4 FCB4303030307800CC- <1>     db  0fch, 0b4h, 030h, 030h, 030h, 030h, 078h, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 0cch, 0fch, 000h
167 00015ACD CCCCCCCCCCFC00      <1>
168 00015AD4 CCCCCCCCCC783000C6- <1>     db  0cch, 0cch, 0cch, 0cch, 0cch, 078h, 030h, 000h, 0c6h, 0c6h, 0c6h, 0d6h, 0feh, 0eeh, 0c6h, 000h
168 00015ADD C6C6D6FEEEC600      <1>
169 00015AE4 C6C66C38386CC600CC- <1>     db  0c6h, 0c6h, 06ch, 038h, 038h, 06ch, 0c6h, 000h, 0cch, 0cch, 0cch, 078h, 030h, 030h, 078h, 000h
169 00015AED CCCC7830307800      <1>
170 00015AF4 FEC68C183266FE0078- <1>     db  0feh, 0c6h, 08ch, 018h, 032h, 066h, 0feh, 000h, 078h, 060h, 060h, 060h, 060h, 060h, 078h, 000h
170 00015AFD 60606060607800      <1>
171 00015B04 C06030180C06020078- <1>     db  0c0h, 060h, 030h, 018h, 00ch, 006h, 002h, 000h, 078h, 018h, 018h, 018h, 018h, 018h, 078h, 000h
171 00015B0D 18181818187800      <1>
172 00015B14 10386CC600000000000- <1>    db  010h, 038h, 06ch, 0c6h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh
172 00015B1D 000000000000FF      <1>
173 00015B24 303018000000000000- <1>     db  030h, 030h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 078h, 00ch, 07ch, 0cch, 076h, 000h
173 00015B2D 00780C7CCC7600      <1>
174 00015B34 E060607C6666DC0000- <1>     db  0e0h, 060h, 060h, 07ch, 066h, 066h, 0dch, 000h, 000h, 000h, 078h, 0cch, 0c0h, 0cch, 078h, 000h
174 00015B3D 0078CCC0CC7800      <1>
175 00015B44 1C0C0C7CCCCC760000- <1>     db  01ch, 00ch, 00ch, 07ch, 0cch, 0cch, 076h, 000h, 000h, 000h, 078h, 0cch, 0fch, 0c0h, 078h, 000h
175 00015B4D 0078CCFCC07800      <1>
176 00015B54 386C60F06060F00000- <1>     db  038h, 06ch, 060h, 0f0h, 060h, 060h, 0f0h, 000h, 000h, 000h, 076h, 0cch, 0cch, 07ch, 00ch, 0f8h
176 00015B5D 0076CCCC7C0CF8      <1>
177 00015B64 E0606C766666E60030- <1>     db  0e0h, 060h, 06ch, 076h, 066h, 066h, 0e6h, 000h, 030h, 000h, 070h, 030h, 030h, 030h, 078h, 000h
177 00015B6D 00703030307800      <1>
178 00015B74 0C000C0C0CCCCC78E0- <1>     db  00ch, 000h, 00ch, 00ch, 00ch, 0cch, 0cch, 078h, 0e0h, 060h, 066h, 06ch, 078h, 06ch, 0e6h, 000h
178 00015B7D 60666C786CE600      <1>
179 00015B84 703030303030780000- <1>     db  070h, 030h, 030h, 030h, 030h, 030h, 078h, 000h, 000h, 000h, 0cch, 0feh, 0feh, 0d6h, 0c6h, 000h
179 00015B8D 00CCFEFED6C600      <1>
180 00015B94 0000F8CCCCCCCC0000- <1>     db  000h, 000h, 0f8h, 0cch, 0cch, 0cch, 0cch, 000h, 000h, 000h, 078h, 0cch, 0cch, 0cch, 078h, 000h
180 00015B9D 0078CCCCCC7800      <1>
181 00015BA4 0000DC66667C60F000- <1>     db  000h, 000h, 0dch, 066h, 066h, 07ch, 060h, 0f0h, 000h, 000h, 076h, 0cch, 0cch, 07ch, 00ch, 01eh
181 00015BAD 0076CCCC7C0C1E      <1>
182 00015BB4 0000DC766660F00000- <1>     db  000h, 000h, 0dch, 076h, 066h, 060h, 0f0h, 000h, 000h, 000h, 07ch, 0c0h, 078h, 00ch, 0f8h, 000h
182 00015BBD 007CC0780CF800      <1>
183 00015BC4 10307C303034180000- <1>     db  010h, 030h, 07ch, 030h, 030h, 034h, 018h, 000h, 000h, 000h, 0cch, 0cch, 0cch, 0cch, 076h, 000h
183 00015BCD 00CCCCCCCC7600      <1>
184 00015BD4 0000CCCCCC78300000- <1>     db  000h, 000h, 0cch, 0cch, 0cch, 078h, 030h, 000h, 000h, 000h, 0c6h, 0d6h, 0feh, 0feh, 06ch, 000h
184 00015BDD 00C6D6FEFE6C00      <1>
185 00015BE4 0000C66C386CC60000- <1>     db  000h, 000h, 0c6h, 06ch, 038h, 06ch, 0c6h, 000h, 000h, 000h, 0cch, 0cch, 0cch, 07ch, 00ch, 0f8h
185 00015BED 00CCCCCC7C0CF8      <1>
186 00015BF4 0000FC983064FC001C- <1>     db  000h, 000h, 0fch, 098h, 030h, 064h, 0fch, 000h, 01ch, 030h, 030h, 0e0h, 030h, 030h, 01ch, 000h
186 00015BFD 3030E030301C00      <1>
187 00015C04 1818180018181800E0- <1>     db  018h, 018h, 018h, 000h, 018h, 018h, 018h, 000h, 0e0h, 030h, 030h, 01ch, 030h, 030h, 0e0h, 000h
187 00015C0D 30301C3030E000      <1>
188 00015C14 76DC00000000000000- <1>     db  076h, 0dch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 010h, 038h, 06ch, 0c6h, 0c6h, 0feh, 000h
188 00015C1D 10386CC6C6FE00      <1>
189 00015C24 78CCC0CC78180C7800- <1>     db  078h, 0cch, 0c0h, 0cch, 078h, 018h, 00ch, 078h, 000h, 0cch, 000h, 0cch, 0cch, 0cch, 07eh, 000h
189 00015C2D CC00CCCCCC7E00      <1>
190 00015C34 1C0078CCFCC078007E- <1>     db  01ch, 000h, 078h, 0cch, 0fch, 0c0h, 078h, 000h, 07eh, 0c3h, 03ch, 006h, 03eh, 066h, 03fh, 000h
190 00015C3D C33C063E663F00      <1>
191 00015C44 CC00780C7CCC7E00E0- <1>     db  0cch, 000h, 078h, 00ch, 07ch, 0cch, 07eh, 000h, 0e0h, 000h, 078h, 00ch, 07ch, 0cch, 07eh, 000h
191 00015C4D 00780C7CCC7E00      <1>
192 00015C54 3030780C7CCC7E0000- <1>     db  030h, 030h, 078h, 00ch, 07ch, 0cch, 07eh, 000h, 000h, 000h, 078h, 0c0h, 0c0h, 078h, 00ch, 038h
192 00015C5D 0078C0C0780C38      <1>
193 00015C64 7EC33C667E603C00CC- <1>     db  07eh, 0c3h, 03ch, 066h, 07eh, 060h, 03ch, 000h, 0cch, 000h, 078h, 0cch, 0fch, 0c0h, 078h, 000h
193 00015C6D 0078CCFCC07800      <1>
194 00015C74 E00078CCFCC07800CC- <1>     db  0e0h, 000h, 078h, 0cch, 0fch, 0c0h, 078h, 000h, 0cch, 000h, 070h, 030h, 030h, 030h, 078h, 000h
194 00015C7D 00703030307800      <1>
195 00015C84 7CC6381818183C00E0- <1>     db  07ch, 0c6h, 038h, 018h, 018h, 018h, 03ch, 000h, 0e0h, 000h, 070h, 030h, 030h, 030h, 078h, 000h
195 00015C8D 00703030307800      <1>
196 00015C94 C6386CC6FEC6C60030- <1>     db  0c6h, 038h, 06ch, 0c6h, 0feh, 0c6h, 0c6h, 000h, 030h, 030h, 000h, 078h, 0cch, 0fch, 0cch, 000h
196 00015C9D 300078CCFCCC00      <1>
197 00015CA4 1C00FC607860FC0000- <1>     db  01ch, 000h, 0fch, 060h, 078h, 060h, 0fch, 000h, 000h, 000h, 07fh, 00ch, 07fh, 0cch, 07fh, 000h
197 00015CAD 007F0C7FCC7F00      <1>
198 00015CB4 3E6CCCFECCCCCE0078- <1>     db  03eh, 06ch, 0cch, 0feh, 0cch, 0cch, 0ceh, 000h, 078h, 0cch, 000h, 078h, 0cch, 0cch, 078h, 000h
198 00015CBD CC0078CCCC7800      <1>
199 00015CC4 00CC0078CCCC780000- <1>     db  000h, 0cch, 000h, 078h, 0cch, 0cch, 078h, 000h, 000h, 0e0h, 000h, 078h, 0cch, 0cch, 078h, 000h
199 00015CCD E00078CCCC7800      <1>
200 00015CD4 78CC00CCCCCC7E0000- <1>     db  078h, 0cch, 000h, 0cch, 0cch, 0cch, 07eh, 000h, 000h, 0e0h, 000h, 0cch, 0cch, 0cch, 07eh, 000h
200 00015CDD E000CCCCCC7E00      <1>
201 00015CE4 00CC00CCCC7C0CF8C3- <1>     db  000h, 0cch, 000h, 0cch, 0cch, 07ch, 00ch, 0f8h, 0c3h, 018h, 03ch, 066h, 066h, 03ch, 018h, 000h
201 00015CED 183C66663C1800      <1>
202 00015CF4 CC00CCCCCCCC780018- <1>     db  0cch, 000h, 0cch, 0cch, 0cch, 0cch, 078h, 000h, 018h, 018h, 07eh, 0c0h, 0c0h, 07eh, 018h, 018h
202 00015CFD 187EC0C07E1818      <1>
203 00015D04 386C64F060E6FC00CC- <1>     db  038h, 06ch, 064h, 0f0h, 060h, 0e6h, 0fch, 000h, 0cch, 0cch, 078h, 0fch, 030h, 0fch, 030h, 030h
203 00015D0D CC78FC30FC3030      <1>
204 00015D14 F8CCCCFAC6CFC6C70E- <1>     db  0f8h, 0cch, 0cch, 0fah, 0c6h, 0cfh, 0c6h, 0c7h, 00eh, 01bh, 018h, 03ch, 018h, 018h, 0d8h, 070h
204 00015D1D 1B183C1818D870      <1>
205 00015D24 1C00780C7CCC7E0038- <1>     db  01ch, 000h, 078h, 00ch, 07ch, 0cch, 07eh, 000h, 038h, 000h, 070h, 030h, 030h, 030h, 078h, 000h
205 00015D2D 00703030307800      <1>
206 00015D34 001C0078CCCC780000- <1>     db  000h, 01ch, 000h, 078h, 0cch, 0cch, 078h, 000h, 000h, 01ch, 000h, 0cch, 0cch, 0cch, 07eh, 000h
206 00015D3D 1C00CCCCCC7E00      <1>
207 00015D44 00F800F8CCCCCC00FC- <1>     db  000h, 0f8h, 000h, 0f8h, 0cch, 0cch, 0cch, 000h, 0fch, 000h, 0cch, 0ech, 0fch, 0dch, 0cch, 000h
207 00015D4D 00CCECFCDCCC00      <1>
208 00015D54 3C6C6C3E007E000038- <1>     db  03ch, 06ch, 06ch, 03eh, 000h, 07eh, 000h, 000h, 038h, 06ch, 06ch, 038h, 000h, 07ch, 000h, 000h
208 00015D5D 6C6C38007C0000      <1>
209 00015D64 30003060C0CC780000- <1>     db  030h, 000h, 030h, 060h, 0c0h, 0cch, 078h, 000h, 000h, 000h, 000h, 0fch, 0c0h, 0c0h, 000h, 000h
209 00015D6D 0000FCC0C00000      <1>
210 00015D74 000000FC0C0C0000C3- <1>     db  000h, 000h, 000h, 0fch, 00ch, 00ch, 000h, 000h, 0c3h, 0c6h, 0cch, 0deh, 033h, 066h, 0cch, 00fh
210 00015D7D C6CCDE3366CC0F      <1>
211 00015D84 C3C6CCDB376FCF0318- <1>     db  0c3h, 0c6h, 0cch, 0dbh, 037h, 06fh, 0cfh, 003h, 018h, 018h, 000h, 018h, 018h, 018h, 018h, 000h
211 00015D8D 18001818181800      <1>
212 00015D94 003366CC6633000000- <1>     db  000h, 033h, 066h, 0cch, 066h, 033h, 000h, 000h, 000h, 0cch, 066h, 033h, 066h, 0cch, 000h, 000h
212 00015D9D CC663366CC0000      <1>
213 00015DA4 228822882288228855- <1>     db  022h, 088h, 022h, 088h, 022h, 088h, 022h, 088h, 055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah
213 00015DAD AA55AA55AA55AA      <1>
214 00015DB4 DB77DBEEDB77DBEE18- <1>     db  0dbh, 077h, 0dbh, 0eeh, 0dbh, 077h, 0dbh, 0eeh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
214 00015DBD 18181818181818      <1>
215 00015DC4 18181818F818181818- <1>     db  018h, 018h, 018h, 018h, 0f8h, 018h, 018h, 018h, 018h, 018h, 0f8h, 018h, 0f8h, 018h, 018h, 018h
215 00015DCD 18F818F8181818      <1>
216 00015DD4 36363636F636363600- <1>     db  036h, 036h, 036h, 036h, 0f6h, 036h, 036h, 036h, 000h, 000h, 000h, 000h, 0feh, 036h, 036h, 036h
216 00015DDD 000000FE363636      <1>
217 00015DE4 0000F818F818181836- <1>     db  000h, 000h, 0f8h, 018h, 0f8h, 018h, 018h, 018h, 036h, 036h, 0f6h, 006h, 0f6h, 036h, 036h, 036h
```

```
217 00015DED 36F606F6363636      <1>
218 00015DF4 363636363636363600- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 000h, 000h, 0feh, 006h, 0f6h, 036h, 036h, 036h
218 00015DFD 00FE06F6363636      <1>
219 00015E04 3636F606FE00000036- <1>     db  036h, 036h, 0f6h, 006h, 0feh, 000h, 000h, 000h, 036h, 036h, 036h, 036h, 0feh, 000h, 000h, 000h
219 00015E0D 363636FE000000      <1>
220 00015E14 1818F818F800000000- <1>     db  018h, 018h, 0f8h, 018h, 0f8h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0f8h, 018h, 018h, 018h
220 00015E1D 000000F8181818      <1>
221 00015E24 181818181F00000018- <1>     db  018h, 018h, 018h, 018h, 01fh, 000h, 000h, 000h, 018h, 018h, 018h, 018h, 0ffh, 000h, 000h, 000h
221 00015E2D 181818FF000000      <1>
222 00015E34 00000000FF18181818- <1>     db  000h, 000h, 000h, 000h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 01fh, 018h, 018h, 018h
222 00015E3D 1818181F181818      <1>
223 00015E44 00000000FF00000018- <1>     db  000h, 000h, 000h, 000h, 0ffh, 000h, 000h, 000h, 018h, 018h, 018h, 018h, 0ffh, 018h, 018h, 018h
223 00015E4D 181818FF181818      <1>
224 00015E54 18181F181F18181836- <1>     db  018h, 018h, 01fh, 018h, 01fh, 018h, 018h, 018h, 036h, 036h, 036h, 036h, 037h, 036h, 036h, 036h
224 00015E5D 36363637363636      <1>
225 00015E64 363637303F00000000- <1>     db  036h, 036h, 037h, 030h, 03fh, 000h, 000h, 000h, 000h, 000h, 03fh, 030h, 037h, 036h, 036h, 036h
225 00015E6D 003F3037363636      <1>
226 00015E74 3636F700FF00000000- <1>     db  036h, 036h, 0f7h, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 0ffh, 000h, 0f7h, 036h, 036h, 036h
226 00015E7D 00FF00F7363636      <1>
227 00015E84 363637303736363600- <1>     db  036h, 036h, 037h, 030h, 037h, 036h, 036h, 036h, 000h, 000h, 0ffh, 000h, 0ffh, 000h, 000h, 000h
227 00015E8D 00FF00FF000000      <1>
228 00015E94 3636F700F736363618- <1>     db  036h, 036h, 0f7h, 000h, 0f7h, 036h, 036h, 036h, 018h, 018h, 0ffh, 000h, 0ffh, 000h, 000h, 000h
228 00015E9D 18FF00FF000000      <1>
229 00015EA4 36363636FF00000000- <1>     db  036h, 036h, 036h, 036h, 0ffh, 000h, 000h, 000h, 000h, 000h, 0ffh, 000h, 0ffh, 018h, 018h, 018h
229 00015EAD 00FF00FF181818      <1>
230 00015EB4 00000000FF36363636- <1>     db  000h, 000h, 000h, 000h, 0ffh, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 03fh, 000h, 000h, 000h
230 00015EBD 3636363F000000      <1>
231 00015EC4 18181F181F00000000- <1>     db  018h, 018h, 01fh, 018h, 01fh, 000h, 000h, 000h, 000h, 000h, 01fh, 018h, 01fh, 018h, 018h, 018h
231 00015ECD 001F181F181818      <1>
232 00015ED4 000000003F36363636- <1>     db  000h, 000h, 000h, 000h, 03fh, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 0ffh, 036h, 036h, 036h
232 00015EDD 363636FF363636      <1>
233 00015EE4 1818FF18FF18181818- <1>     db  018h, 018h, 0ffh, 018h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 0f8h, 000h, 000h, 000h
233 00015EED 181818F8000000      <1>
234 00015EF4 000000001F181818FF- <1>     db  000h, 000h, 000h, 000h, 01fh, 018h, 018h, 018h, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh
234 00015EFD FFFFFFFFFFFFFF      <1>
235 00015F04 00000000FFFFFFFFF0- <1>     db  000h, 000h, 000h, 000h, 0ffh, 0ffh, 0ffh, 0ffh, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h
235 00015F0D F0F0F0F0F0F0F0      <1>
236 00015F14 0F0F0F0F0F0F0F0FFF- <1>     db  00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 0ffh, 0ffh, 0ffh, 0ffh, 000h, 000h, 000h, 000h
236 00015F1D FFFFFF00000000      <1>
237 00015F24 000076DCC8DC760000- <1>     db  000h, 000h, 076h, 0dch, 0c8h, 0dch, 076h, 000h, 000h, 078h, 0cch, 0f8h, 0cch, 0f8h, 0c0h, 0c0h
237 00015F2D 78CCF8CCF8C0C0      <1>
238 00015F34 00FCCCC0C0C0C00000- <1>     db  000h, 0fch, 0cch, 0c0h, 0c0h, 0c0h, 0c0h, 000h, 000h, 0feh, 06ch, 06ch, 06ch, 06ch, 06ch, 000h
238 00015F3D FE6C6C6C6C6C00      <1>
239 00015F44 FCCC603060CCFC0000- <1>     db  0fch, 0cch, 060h, 030h, 060h, 0cch, 0fch, 000h, 000h, 000h, 07eh, 0d8h, 0d8h, 0d8h, 070h, 000h
239 00015F4D 007ED8D8D87000      <1>
240 00015F54 00666666667C60C000- <1>     db  000h, 066h, 066h, 066h, 066h, 07ch, 060h, 0c0h, 000h, 076h, 0dch, 018h, 018h, 018h, 018h, 000h
240 00015F5D 76DC1818181800      <1>
241 00015F64 FC3078CCCC7830FC38- <1>     db  0fch, 030h, 078h, 0cch, 0cch, 078h, 030h, 0fch, 038h, 06ch, 0c6h, 0feh, 0c6h, 06ch, 038h, 000h
241 00015F6D 6CC6FEC66C3800      <1>
242 00015F74 386CC6C66C6CEE001C- <1>     db  038h, 06ch, 0c6h, 0c6h, 06ch, 06ch, 0eeh, 000h, 01ch, 030h, 018h, 07ch, 0cch, 0cch, 078h, 000h
242 00015F7D 30187CCCCC7800      <1>
243 00015F84 00007EDBDB7E000006- <1>     db  000h, 000h, 07eh, 0dbh, 0dbh, 07eh, 000h, 000h, 006h, 00ch, 07eh, 0dbh, 0dbh, 07eh, 060h, 0c0h
243 00015F8D 0C7EDBDB7E60C0      <1>
244 00015F94 3860C0F8C060380078- <1>     db  038h, 060h, 0c0h, 0f8h, 0c0h, 060h, 038h, 000h, 078h, 0cch, 0cch, 0cch, 0cch, 0cch, 0cch, 000h
244 00015F9D CCCCCCCCCCCC00      <1>
245 00015FA4 00FC00FC00FC000030- <1>     db  000h, 0fch, 000h, 0fch, 000h, 0fch, 000h, 000h, 030h, 030h, 0fch, 030h, 030h, 000h, 0fch, 000h
245 00015FAD 30FC303000FC00      <1>
246 00015FB4 603018306000FC0018- <1>     db  060h, 030h, 018h, 030h, 060h, 000h, 0fch, 000h, 018h, 030h, 060h, 030h, 018h, 000h, 0fch, 000h
246 00015FBD 3060301800FC00      <1>
247 00015FC4 0E1B1B181818181818- <1>     db  00eh, 01bh, 01bh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 0d8h, 0d8h, 070h
247 00015FCD 18181818D8D870      <1>
248 00015FD4 303000FC00303000000- <1>     db  030h, 030h, 000h, 0fch, 000h, 030h, 030h, 000h, 000h, 076h, 0dch, 000h, 076h, 0dch, 000h, 000h
248 00015FDD 76DC0076DC0000      <1>
249 00015FE4 386C6C380000000000- <1>     db  038h, 06ch, 06ch, 038h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 000h, 000h, 000h
249 00015FED 00001818000000      <1>
250 00015FF4 000000001800000000F- <1>     db  000h, 000h, 000h, 000h, 018h, 000h, 000h, 000h, 00fh, 00ch, 00ch, 00ch, 0ech, 06ch, 03ch, 01ch
250 00015FFD 0C0C0CEC6C3C1C      <1>
251 00016004 786C6C6C6C00000070- <1>     db  078h, 06ch, 06ch, 06ch, 06ch, 000h, 000h, 000h, 070h, 018h, 030h, 060h, 078h, 000h, 000h, 000h
251 0001600D 18306078000000      <1>
252 00016014 00003C3C3C3C000000- <1>     db  000h, 000h, 03ch, 03ch, 03ch, 03ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
252 0001601D 00000000000000      <1>
253                              <1> vgafont14:
254 00016024 000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
254 0001602D 00000000000000      <1>
255 00016034 7E81A58181BD99817E- <1>     db  07eh, 081h, 0a5h, 081h, 081h, 0bdh, 099h, 081h, 07eh, 000h, 000h, 000h, 000h, 000h, 07eh, 0ffh
255 0001603D 00000000007EFF      <1>
256 00016044 DBFFFFC3E7FF7E0000- <1>     db  0dbh, 0ffh, 0ffh, 0c3h, 0e7h, 0ffh, 07eh, 000h, 000h, 000h, 000h, 000h, 000h, 06ch, 0feh, 0feh
256 0001604D 000000006CFEFE      <1>
257 00016054 FEFE7C381000000000- <1>     db  0feh, 0feh, 07ch, 038h, 010h, 000h, 000h, 000h, 000h, 000h, 000h, 010h, 038h, 07ch, 0feh, 07ch
257 0001605D 000010387CFE7C      <1>
258 00016064 3810000000000000018- <1>     db  038h, 010h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 03ch, 03ch, 0e7h, 0e7h, 0e7h, 018h, 018h
258 0001606D 3C3CE7E7E71818      <1>
259 00016074 3C0000000000183C7E- <1>     db  03ch, 000h, 000h, 000h, 000h, 000h, 018h, 03ch, 07eh, 0ffh, 0ffh, 07eh, 018h, 018h, 03ch, 000h
259 0001607D FFFF7E18183C00      <1>
260 00016084 00000000000000183C- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 03ch, 03ch, 018h, 000h, 000h, 000h, 000h, 000h
260 0001608D 3C180000000000      <1>
261 00016094 FFFFFFFFFFE7C3C3E7- <1>     db  0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0e7h, 0c3h, 0c3h, 0e7h, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 000h, 000h
261 0001609D FFFFFFFFFF0000      <1>
262 000160A4 00003C664242663C00- <1>     db  000h, 000h, 03ch, 066h, 042h, 042h, 066h, 03ch, 000h, 000h, 000h, 000h, 0ffh, 0ffh, 0ffh, 0ffh
262 000160AD 000000FFFFFFFF      <1>
263 000160B4 C399BDBD99C3FFFFFF- <1>     db  0c3h, 099h, 0bdh, 0bdh, 099h, 0c3h, 0ffh, 0ffh, 0ffh, 0ffh, 000h, 000h, 01eh, 00eh, 01ah, 032h
263 000160BD FF00001E0E1A32      <1>
264 000160C4 78CCCCCC7800000000- <1>     db  078h, 0cch, 0cch, 0cch, 078h, 000h, 000h, 000h, 000h, 000h, 03ch, 066h, 066h, 066h, 03ch, 018h
264 000160CD 003C6666663C18      <1>
265 000160D4 7E1818000000000003F- <1>     db  07eh, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 03fh, 033h, 03fh, 030h, 030h, 030h, 070h, 0f0h
265 000160DD 333F30303070F0      <1>
266 000160E4 E000000000007F637F- <1>     db  0e0h, 000h, 000h, 000h, 000h, 000h, 07fh, 063h, 07fh, 063h, 063h, 063h, 067h, 0e7h, 0e6h, 0c0h
266 000160ED 63636367E7E6C0      <1>
267 000160F4 0000000001818DB3CE7- <1>     db  000h, 000h, 000h, 000h, 018h, 018h, 0dbh, 03ch, 0e7h, 03ch, 0dbh, 018h, 018h, 000h, 000h, 000h
267 000160FD 3CDB1818000000      <1>
268 00016104 000080C0E0F8FEF8E0- <1>     db  000h, 000h, 080h, 0c0h, 0e0h, 0f8h, 0feh, 0f8h, 0e0h, 0c0h, 080h, 000h, 000h, 000h, 000h, 000h
268 0001610D C0800000000000      <1>
269 00016114 02060E3EFE3E0E0602- <1>     db  002h, 006h, 00eh, 03eh, 0feh, 03eh, 00eh, 006h, 002h, 000h, 000h, 000h, 000h, 000h, 018h, 03ch
269 0001611D 0000000000183C      <1>
270 00016124 7E1818187E3C180000- <1>     db  07eh, 018h, 018h, 018h, 07eh, 03ch, 018h, 000h, 000h, 000h, 000h, 000h, 066h, 066h, 066h, 066h
270 0001612D 00000066666666      <1>
271 00016134 666600666600000000- <1>     db  066h, 066h, 000h, 066h, 066h, 000h, 000h, 000h, 000h, 000h, 07fh, 0dbh, 0dbh, 0dbh, 07bh, 01bh
271 0001613D 007FDBDBDB7B1B      <1>
272 00016144 1B1B1B000000007CC6- <1>     db  01bh, 01bh, 01bh, 000h, 000h, 000h, 000h, 07ch, 0c6h, 060h, 038h, 06ch, 0c6h, 0c6h, 06ch, 038h
272 0001614D 60386CC6C66C38      <1>
273 00016154 0CC67C000000000000- <1>     db  00ch, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0feh, 0feh, 0feh, 000h
273 0001615D 000000FEFEFE00      <1>
274 00016164 00000000183C7E1818- <1>     db  000h, 000h, 000h, 000h, 018h, 03ch, 07eh, 018h, 018h, 018h, 07eh, 03ch, 018h, 07eh, 000h, 000h
274 0001616D 187E3C187E0000      <1>
275 00016174 0000183C7E18181818- <1>     db  000h, 000h, 018h, 03ch, 07eh, 018h, 018h, 018h, 018h, 018h, 018h, 000h, 000h, 000h, 000h, 000h
275 0001617D 18180000000000      <1>
276 00016184 1818181818187E3C18- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 07eh, 03ch, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
276 0001618D 00000000000000      <1>
```

```
277 00016194 180CFE0C1800000000- <1>     db  018h, 00ch, 0feh, 00ch, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 030h, 060h
277 0001619D 00000000003060      <1>
278 000161A4 FE6030000000000000- <1>     db  0feh, 060h, 030h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0c0h, 0c0h, 0c0h
278 000161AD 00000000C0C0C0      <1>
279 000161B4 FE0000000000000000- <1>     db  0feh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 028h, 06ch, 0feh, 06ch, 028h, 000h
279 000161BD 00286CFE6C2800      <1>
280 000161C4 000000000000001038- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 010h, 038h, 038h, 07ch, 07ch, 0feh, 0feh, 000h, 000h
280 000161CD 387C7CFEFE0000      <1>
281 000161D4 0000000000FEFE7C7C- <1>     db  000h, 000h, 000h, 000h, 000h, 0feh, 0feh, 07ch, 07ch, 038h, 038h, 010h, 000h, 000h, 000h, 000h
281 000161DD 38381000000000      <1>
282 000161E4 000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
282 000161ED 00000000000000      <1>
283 000161F4 183C3C3C1818001818- <1>     db  018h, 03ch, 03ch, 03ch, 018h, 018h, 000h, 018h, 018h, 000h, 000h, 000h, 000h, 066h, 066h, 066h
283 000161FD 00000000666666      <1>
284 00016204 240000000000000000- <1>     db  024h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 06ch, 06ch, 0feh, 06ch
284 0001620D 0000006C6CFE6C      <1>
285 00016214 6C6CFE6C6C00000018- <1>     db  06ch, 06ch, 0feh, 06ch, 06ch, 000h, 000h, 000h, 018h, 018h, 07ch, 0c6h, 0c2h, 0c0h, 07ch, 006h
285 0001621D 187CC6C2C07C06      <1>
286 00016224 86C67C181800000000- <1>     db  086h, 0c6h, 07ch, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 0c2h, 0c6h, 00ch, 018h, 030h, 066h
286 0001622D 00C2C60C183066      <1>
287 00016234 C600000000000386C6C- <1>     db  0c6h, 000h, 000h, 000h, 000h, 000h, 038h, 06ch, 06ch, 038h, 076h, 0dch, 0cch, 0cch, 076h, 000h
287 0001623D 3876DCCCCC7600      <1>
288 00016244 000000303030600000- <1>     db  000h, 000h, 000h, 030h, 030h, 030h, 060h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
288 0001624D 00000000000000      <1>
289 00016254 00000C183030303030- <1>     db  000h, 000h, 00ch, 018h, 030h, 030h, 030h, 030h, 030h, 018h, 00ch, 000h, 000h, 000h, 000h, 000h
289 0001625D 180C0000000000      <1>
290 00016264 30180C0C0C0C0C1830- <1>     db  030h, 018h, 00ch, 00ch, 00ch, 00ch, 00ch, 018h, 030h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
290 0001626D 00000000000000      <1>
291 00016274 663CFF3C6600000000- <1>     db  066h, 03ch, 0ffh, 03ch, 066h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h
291 0001627D 00000000001818      <1>
292 00016284 7E1818000000000000- <1>     db  07eh, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
292 0001628D 00000000000000      <1>
293 00016294 181818300000000000- <1>     db  018h, 018h, 018h, 030h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0feh, 000h, 000h, 000h
293 0001629D 000000FE000000      <1>
294 000162A4 000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 000h
294 000162AD 00000000181800      <1>
295 000162B4 0000000002060C1830- <1>     db  000h, 000h, 000h, 000h, 002h, 006h, 00ch, 018h, 030h, 060h, 0c0h, 080h, 000h, 000h, 000h, 000h
295 000162BD 60C08000000000      <1>
296 000162C4 00007CC6CEDEF6E6C6- <1>     db  000h, 000h, 07ch, 0c6h, 0ceh, 0deh, 0f6h, 0e6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h
296 000162CD C67C0000000000      <1>
297 000162D4 183878181818187E- <1>     db  018h, 038h, 078h, 018h, 018h, 018h, 018h, 018h, 07eh, 000h, 000h, 000h, 000h, 000h, 07ch, 0c6h
297 000162DD 00000000007CC6      <1>
298 000162E4 060C183060C6FE0000- <1>     db  006h, 00ch, 018h, 030h, 060h, 0c6h, 0feh, 000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 006h, 006h
298 000162ED 0000007CC60606      <1>
299 000162F4 3C0606C67C00000000- <1>     db  03ch, 006h, 006h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h, 00ch, 01ch, 03ch, 06ch, 0cch, 0feh
299 000162FD 000C1C3C6CCCFE      <1>
300 00016304 0C0C1E0000000000FE- <1>     db  00ch, 00ch, 01eh, 000h, 000h, 000h, 000h, 000h, 0feh, 0c0h, 0c0h, 0c0h, 0fch, 006h, 006h, 0c6h
300 0001630D C0C0C0FC0606C6      <1>
301 00016314 7C00000000003860C0- <1>     db  07ch, 000h, 000h, 000h, 000h, 000h, 038h, 060h, 0c0h, 0c0h, 0fch, 0c6h, 0c6h, 0c6h, 07ch, 000h
301 0001631D C0FCC6C6C67C00      <1>
302 00016324 00000000FEC6060C18- <1>     db  000h, 000h, 000h, 000h, 0feh, 0c6h, 006h, 00ch, 018h, 030h, 030h, 030h, 030h, 000h, 000h, 000h
302 0001632D 30303030000000      <1>
303 00016334 00007CC6C6C67CC6C6- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 07ch, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h
303 0001633D C67C0000000000      <1>
304 00016344 7CC6C6C67E06060C78- <1>     db  07ch, 0c6h, 0c6h, 0c6h, 07eh, 006h, 006h, 00ch, 078h, 000h, 000h, 000h, 000h, 000h, 000h, 018h
304 0001634D 00000000000018      <1>
305 00016354 180000001818000000- <1>     db  018h, 000h, 000h, 000h, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 000h
305 0001635D 00000000181800      <1>
306 00016364 000018183000000000- <1>     db  000h, 000h, 018h, 018h, 030h, 000h, 000h, 000h, 000h, 000h, 006h, 00ch, 018h, 030h, 060h, 030h
306 0001636D 00060C18306030      <1>
307 00016374 180C06000000000000- <1>     db  018h, 00ch, 006h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 07eh, 000h, 000h, 07eh, 000h
307 0001637D 00007E00007E00      <1>
308 00016384 000000000000603018- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 060h, 030h, 018h, 00ch, 006h, 00ch, 018h, 030h, 060h, 000h
308 0001638D 0C060C18306000      <1>
309 00016394 000000007CC6C60C18- <1>     db  000h, 000h, 000h, 000h, 07ch, 0c6h, 0c6h, 00ch, 018h, 018h, 000h, 018h, 018h, 000h, 000h, 000h
309 0001639D 18001818000000      <1>
310 000163A4 00007CC6C6DEDEDEDC- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 0deh, 0deh, 0deh, 0dch, 0c0h, 07ch, 000h, 000h, 000h, 000h, 000h
310 000163AD C07C0000000000      <1>
311 000163B4 10386CC6C6FEC6C6C6- <1>     db  010h, 038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h, 000h, 000h, 0fch, 066h
311 000163BD 0000000000FC66      <1>
312 000163C4 66667C666666FC0000- <1>     db  066h, 066h, 07ch, 066h, 066h, 066h, 0fch, 000h, 000h, 000h, 000h, 000h, 03ch, 066h, 0c2h, 0c0h
312 000163CD 0000003C66C2C0      <1>
313 000163D4 C0C0C2663C00000000- <1>     db  0c0h, 0c0h, 0c2h, 066h, 03ch, 000h, 000h, 000h, 000h, 000h, 0f8h, 06ch, 066h, 066h, 066h, 066h
313 000163DD 00F86C66666666      <1>
314 000163E4 666CF800000000000FE- <1>     db  066h, 06ch, 0f8h, 000h, 000h, 000h, 000h, 000h, 0feh, 066h, 062h, 068h, 078h, 068h, 062h, 066h
314 000163ED 66626878686266      <1>
315 000163F4 FE00000000000FE6662- <1>     db  0feh, 000h, 000h, 000h, 000h, 000h, 0feh, 066h, 062h, 068h, 078h, 068h, 060h, 060h, 0f0h, 000h
315 000163FD 6878686060F000      <1>
316 00016404 000000003C66C2C0C0- <1>     db  000h, 000h, 000h, 000h, 03ch, 066h, 0c2h, 0c0h, 0c0h, 0deh, 0c6h, 066h, 03ah, 000h, 000h, 000h
316 0001640D DEC6663A000000      <1>
317 00016414 0000C6C6C6C6FEC6C6- <1>     db  000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h, 000h, 000h
317 0001641D C6C600000000000     <1>
318 00016424 3C1818181818183C- <1>     db  03ch, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h, 000h, 01eh, 00ch
318 0001642D 00000000001E0C      <1>
319 00016434 0C0C0C0CCCCCC780000- <1>     db  00ch, 00ch, 00ch, 00ch, 0cch, 0cch, 078h, 000h, 000h, 000h, 000h, 000h, 0e6h, 066h, 06ch, 06ch
319 0001643D 000000E6666C6C      <1>
320 00016444 786C6C66E600000000- <1>     db  078h, 06ch, 06ch, 066h, 0e6h, 000h, 000h, 000h, 000h, 000h, 0f0h, 060h, 060h, 060h, 060h, 060h
320 0001644D 00F06060606060      <1>
321 00016454 6266FE0000000000C6- <1>     db  062h, 066h, 0feh, 000h, 000h, 000h, 000h, 000h, 0c6h, 0eeh, 0feh, 0feh, 0d6h, 0c6h, 0c6h, 0c6h
321 0001645D EEFEFED6C6C6C6      <1>
322 00016464 C600000000000C6E6F6- <1>     db  0c6h, 000h, 000h, 000h, 000h, 000h, 0c6h, 0e6h, 0f6h, 0feh, 0deh, 0ceh, 0c6h, 0c6h, 0c6h, 000h
322 0001646D FEDECEC6C6C600      <1>
323 00016474 00000000386CC6C6C6- <1>     db  000h, 000h, 000h, 000h, 038h, 06ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 06ch, 038h, 000h, 000h, 000h
323 0001647D C6C66C38000000      <1>
324 00016484 0000FC6666667C6060- <1>     db  000h, 000h, 0fch, 066h, 066h, 066h, 07ch, 060h, 060h, 060h, 0f0h, 000h, 000h, 000h, 000h, 000h
324 0001648D 60F00000000000      <1>
325 00016494 7CC6C6C6C6D6DE7C0C- <1>     db  07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0d6h, 0deh, 07ch, 00ch, 00eh, 000h, 000h, 000h, 000h, 0fch, 066h
325 0001649D 0E00000000FC66      <1>
326 000164A4 66667C6C6666E60000- <1>     db  066h, 066h, 07ch, 06ch, 066h, 066h, 0e6h, 000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 0c6h, 060h
326 000164AD 0000007CC6C660      <1>
327 000164B4 380CC6C67C00000000- <1>     db  038h, 00ch, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h, 07eh, 07eh, 05ah, 018h, 018h, 018h
327 000164BD 007E7E5A181818      <1>
328 000164C4 18183C0000000000C6- <1>     db  018h, 018h, 03ch, 000h, 000h, 000h, 000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h
328 000164CD C6C6C6C6C6C6C6      <1>
329 000164D4 7C0000000000C6C6C6- <1>     db  07ch, 000h, 000h, 000h, 000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 06ch, 038h, 010h, 000h
329 000164DD C6C6C66C381000      <1>
330 000164E4 00000000C6C6C6C6D6- <1>     db  000h, 000h, 000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0d6h, 0d6h, 0feh, 07ch, 06ch, 000h, 000h, 000h
330 000164ED D6FE7C6C000000      <1>
331 000164F4 0000C6C66C3838386C- <1>     db  000h, 000h, 0c6h, 0c6h, 06ch, 038h, 038h, 038h, 06ch, 0c6h, 0c6h, 000h, 000h, 000h, 000h, 000h
331 000164FD C6C600000000000     <1>
332 00016504 666666663C1818183C- <1>     db  066h, 066h, 066h, 066h, 03ch, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h, 000h, 0feh, 0c6h
332 0001650D 0000000000FEC6      <1>
333 00016514 8C183060C2C6FE0000- <1>     db  08ch, 018h, 030h, 060h, 0c2h, 0c6h, 0feh, 000h, 000h, 000h, 000h, 000h, 03ch, 030h, 030h, 030h
333 0001651D 0000003C303030      <1>
334 00016524 303030303C00000000- <1>     db  030h, 030h, 030h, 030h, 03ch, 000h, 000h, 000h, 000h, 000h, 080h, 0c0h, 0e0h, 070h, 038h, 01ch
334 0001652D 0080C0E070381C      <1>
335 00016534 0E060200000000003C- <1>     db  00eh, 006h, 002h, 000h, 000h, 000h, 000h, 000h, 03ch, 00ch, 00ch, 00ch, 00ch, 00ch, 00ch, 00ch
335 0001653D 0C0C0C0C0C0C0C      <1>
```

```
336 00016544 3C00000010386CC600- <1>     db  03ch, 000h, 000h, 000h, 010h, 038h, 06ch, 0c6h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
336 0001654D 00000000000000      <1>
337 00016554 00000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh, 000h
337 0001655D 0000000000FF00      <1>
338 00016564 303018000000000000- <1>     db  030h, 030h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
338 0001656D 00000000000000      <1>
339 00016574 000000780C7CCCCC76- <1>     db  000h, 000h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h, 000h, 0e0h, 060h
339 0001657D 0000000000E060      <1>
340 00016584 60786C6666667C0000- <1>     db  060h, 078h, 06ch, 066h, 066h, 066h, 07ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 07ch
340 0001658D 0000000000007C      <1>
341 00016594 C6C0C0C67C00000000- <1>     db  0c6h, 0c0h, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h, 01ch, 00ch, 00ch, 03ch, 06ch, 0cch
341 0001659D 001C0C0C3C6CCC      <1>
342 000165A4 CCCC76000000000000- <1>     db  0cch, 0cch, 076h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 0feh, 0c0h, 0c6h
342 000165AD 00007CC6FEC0C6      <1>
343 000165B4 7C0000000000386C64- <1>     db  07ch, 000h, 000h, 000h, 000h, 000h, 038h, 06ch, 064h, 060h, 0f0h, 060h, 060h, 060h, 0f0h, 000h
343 000165BD 60F0606060F000      <1>
344 000165C4 0000000000000076CC- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 076h, 0cch, 0cch, 0cch, 07ch, 00ch, 0cch, 078h, 000h
344 000165CD CCCC7C0CCC7800      <1>
345 000165D4 0000E060606C766666- <1>     db  000h, 000h, 0e0h, 060h, 060h, 06ch, 076h, 066h, 066h, 066h, 0e6h, 000h, 000h, 000h, 000h, 000h
345 000165DD 66E60000000000      <1>
346 000165E4 18180038181818183C- <1>     db  018h, 018h, 000h, 038h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h, 000h, 006h, 006h
346 000165ED 00000000000606      <1>
347 000165F4 000E0606060666663C- <1>     db  000h, 00eh, 006h, 006h, 006h, 006h, 066h, 066h, 03ch, 000h, 000h, 000h, 0e0h, 060h, 060h, 066h
347 000165FD 000000E0606066      <1>
348 00016604 6C786C66E600000000- <1>     db  06ch, 078h, 06ch, 066h, 0e6h, 000h, 000h, 000h, 000h, 000h, 038h, 018h, 018h, 018h, 018h, 018h
348 0001660D 00381818181818      <1>
349 00016614 18183C000000000000- <1>     db  018h, 018h, 03ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ech, 0feh, 0d6h, 0d6h, 0d6h
349 0001661D 0000ECFED6D6D6      <1>
350 00016624 C600000000000000000- <1>     db  0c6h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0dch, 066h, 066h, 066h, 066h, 066h, 000h
350 0001662D DC666666666600      <1>
351 00016634 000000000000007CC6- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h
351 0001663D C6C6C67C000000      <1>
352 00016644 0000000000DC666666- <1>     db  000h, 000h, 000h, 000h, 000h, 0dch, 066h, 066h, 066h, 07ch, 060h, 060h, 0f0h, 000h, 000h, 000h
352 0001664D 7C6060F0000000      <1>
353 00016654 00000076CCCCCC7C0C- <1>     db  000h, 000h, 000h, 076h, 0cch, 0cch, 0cch, 07ch, 00ch, 00ch, 01eh, 000h, 000h, 000h, 000h, 000h
353 0001665D 0C1E0000000000      <1>
354 00016664 00DC76666060F00000- <1>     db  000h, 0dch, 076h, 066h, 060h, 060h, 0f0h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 07ch
354 0001666D 0000000000007C      <1>
355 00016674 C6701CC67C00000000- <1>     db  0c6h, 070h, 01ch, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h, 010h, 030h, 030h, 0fch, 030h, 030h
355 0001667D 00103030FC3030      <1>
356 00016684 30361C000000000000- <1>     db  030h, 036h, 01ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0cch, 0cch, 0cch, 0cch, 0cch
356 0001668D 0000CCCCCCCCCC      <1>
357 00016694 760000000000000000- <1>     db  076h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 066h, 066h, 066h, 066h, 03ch, 018h, 000h
357 0001669D 666666663C1800      <1>
358 000166A4 00000000000000C6C6- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0c6h, 0c6h, 0d6h, 0d6h, 0feh, 06ch, 000h, 000h, 000h
358 000166AD D6D6FE6C000000      <1>
359 000166B4 0000000000C66C3838- <1>     db  000h, 000h, 000h, 000h, 000h, 0c6h, 06ch, 038h, 038h, 06ch, 0c6h, 000h, 000h, 000h, 000h, 000h
359 000166BD 6CC600000000000     <1>
360 000166C4 000000C6C6C6C67E06- <1>     db  000h, 000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 07eh, 006h, 00ch, 0f8h, 000h, 000h, 000h, 000h, 000h
360 000166CD 0CF80000000000      <1>
361 000166D4 00FECC183066FE0000- <1>     db  000h, 0feh, 0cch, 018h, 030h, 066h, 0feh, 000h, 000h, 000h, 000h, 000h, 00eh, 018h, 018h, 018h
361 000166DD 0000000E181818      <1>
362 000166E4 701818180E00000000- <1>     db  070h, 018h, 018h, 018h, 00eh, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 018h, 018h, 000h, 018h
362 000166ED 00181818180018      <1>
363 000166F4 1818180000000000070- <1>     db  018h, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 070h, 018h, 018h, 018h, 00eh, 018h, 018h, 018h
363 000166FD 1818180E181818      <1>
364 00016704 70000000000076DC00- <1>     db  070h, 000h, 000h, 000h, 000h, 000h, 076h, 0dch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
364 0001670D 00000000000000      <1>
365 00016714 000000000000010386C- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 010h, 038h, 06ch, 0c6h, 0c6h, 0feh, 000h, 000h, 000h, 000h
365 0001671D C6C6FE00000000      <1>
366 00016724 00003C66C2C0C0C266- <1>     db  000h, 000h, 03ch, 066h, 0c2h, 0c0h, 0c0h, 0c2h, 066h, 03ch, 00ch, 006h, 07ch, 000h, 000h, 000h
366 0001672D 3C0C067C000000      <1>
367 00016734 CCCC00CCCCCCCCCC76- <1>     db  0cch, 0cch, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h, 00ch, 018h, 030h
367 0001673D 000000000C1830      <1>
368 00016744 007CC6FEC0C67C0000- <1>     db  000h, 07ch, 0c6h, 0feh, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 010h, 038h, 06ch, 000h, 078h
368 0001674D 000010386C0078      <1>
369 00016754 0C7CCCCC7600000000- <1>     db  00ch, 07ch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h, 000h, 0cch, 0cch, 000h, 078h, 00ch, 07ch
369 0001675D 00CCCC00780C7C      <1>
370 00016764 CCCC76000000006030- <1>     db  0cch, 0cch, 076h, 000h, 000h, 000h, 000h, 060h, 030h, 018h, 000h, 078h, 00ch, 07ch, 0cch, 0cch
370 0001676D 1800780C7CCCCC      <1>
371 00016774 7600000000386C3800- <1>     db  076h, 000h, 000h, 000h, 000h, 038h, 06ch, 038h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 076h, 000h
371 0001677D 780C7CCCCC7600      <1>
372 00016784 0000000000003C6660- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 03ch, 066h, 060h, 066h, 03ch, 00ch, 006h, 03ch, 000h, 000h
372 0001678D 663C0C063C0000      <1>
373 00016794 0010386C007CC6FEC0- <1>     db  000h, 010h, 038h, 06ch, 000h, 07ch, 0c6h, 0feh, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h
373 0001679D C67C0000000000      <1>
374 000167A4 CCCC007CC6FEC0C67C- <1>     db  0cch, 0cch, 000h, 07ch, 0c6h, 0feh, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 060h, 030h, 018h
374 000167AD 00000000603018      <1>
375 000167B4 007CC6FEC0C67C0000- <1>     db  000h, 07ch, 0c6h, 0feh, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h, 066h, 066h, 000h, 038h
375 000167BD 00000066660038      <1>
376 000167C4 181818183C00000000- <1>     db  018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h, 018h, 03ch, 066h, 000h, 038h, 018h, 018h
376 000167CD 183C6600381818      <1>
377 000167D4 18183C000000006030- <1>     db  018h, 018h, 03ch, 000h, 000h, 000h, 000h, 060h, 030h, 018h, 000h, 038h, 018h, 018h, 018h, 018h
377 000167DD 18003818181818      <1>
378 000167E4 3C00000000C6C61038- <1>     db  03ch, 000h, 000h, 000h, 000h, 0c6h, 0c6h, 010h, 038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 000h
378 000167ED 6CC6C6FEC6C600      <1>
379 000167F4 0000386C3800386CC6- <1>     db  000h, 000h, 038h, 06ch, 038h, 000h, 038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 000h, 000h, 000h
379 000167FD C6FEC6C6000000      <1>
380 00016804 18306000FE66607C60- <1>     db  018h, 030h, 060h, 000h, 0feh, 066h, 060h, 07ch, 060h, 066h, 0feh, 000h, 000h, 000h, 000h, 000h
380 0001680D 66FE0000000000      <1>
381 00016814 0000CC76367ED8D86E- <1>     db  000h, 000h, 0cch, 076h, 036h, 07eh, 0d8h, 0d8h, 06eh, 000h, 000h, 000h, 000h, 000h, 03eh, 06ch
381 0001681D 000000000003E6C     <1>
382 00016824 CCCCFECCCCCCCE0000- <1>     db  0cch, 0cch, 0feh, 0cch, 0cch, 0cch, 0ceh, 000h, 000h, 000h, 000h, 010h, 038h, 06ch, 000h, 07ch
382 0001682D 000010386C007C      <1>
383 00016834 C6C6C6C67C00000000- <1>     db  0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h, 0c6h, 0c6h, 000h, 07ch, 0c6h, 0c6h
383 0001683D 00C6C6007CC6C6      <1>
384 00016844 C6C67C000000006030- <1>     db  0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 060h, 030h, 018h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h
384 0001684D 18007CC6C6C6C6      <1>
385 00016854 7C0000000003078CC00- <1>     db  07ch, 000h, 000h, 000h, 000h, 030h, 078h, 0cch, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 076h, 000h
385 0001685D CCCCCCCCCC7600      <1>
386 00016864 00000060301800CCCC- <1>     db  000h, 000h, 000h, 060h, 030h, 018h, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h
386 0001686D CCCCCC76000000      <1>
387 00016874 0000C6C600C6C6C6C6- <1>     db  000h, 000h, 0c6h, 0c6h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 07eh, 006h, 00ch, 078h, 000h, 000h, 0c6h
387 0001687D 7E060C780000C6      <1>
388 00016884 C6386CC6C6C6C66C38- <1>     db  0c6h, 038h, 06ch, 0c6h, 0c6h, 0c6h, 0c6h, 06ch, 038h, 000h, 000h, 000h, 000h, 0c6h, 0c6h, 000h
388 0001688D 00000000C6C600      <1>
389 00016894 C6C6C6C6C6C67C0000- <1>     db  0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 018h, 018h, 03ch, 066h, 060h
389 0001689D 000018183C6660      <1>
390 000168A4 60663C181800000000- <1>     db  060h, 066h, 03ch, 018h, 018h, 000h, 000h, 000h, 000h, 038h, 06ch, 064h, 060h, 0f0h, 060h, 060h
390 000168AD 386C6460F06060      <1>
391 000168B4 60E6FC000000000066- <1>     db  060h, 0e6h, 0fch, 000h, 000h, 000h, 000h, 000h, 066h, 066h, 03ch, 018h, 07eh, 018h, 07eh, 018h
391 000168BD 663C187E187E18      <1>
392 000168C4 1800000000F8CCCCF8- <1>     db  018h, 000h, 000h, 000h, 000h, 0f8h, 0cch, 0cch, 0f8h, 0c4h, 0cch, 0deh, 0cch, 0cch, 0c6h, 000h
392 000168CD C4CCDECCCCC600      <1>
393 000168D4 0000000E1B1818187E- <1>     db  000h, 000h, 000h, 00eh, 01bh, 018h, 018h, 018h, 07eh, 018h, 018h, 018h, 018h, 0d8h, 070h, 000h
393 000168DD 18181818D87000      <1>
394 000168E4 0018306000780C7CCC- <1>     db  000h, 018h, 030h, 060h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h, 00ch
394 000168ED CC76000000000C      <1>
```

```
395 000168F4 18300038181818183C- <1>     db  018h, 030h, 000h, 038h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h, 018h, 030h, 060h
395 000168FD 00000000183060      <1>
396 00016904 007CC6C6C6C67C0000- <1>     db  000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 018h, 030h, 060h, 000h, 0cch
396 0001690D 000018306000CC      <1>
397 00016914 CCCCCCCC7600000000- <1>     db  0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h, 000h, 076h, 0dch, 000h, 0dch, 066h, 066h
397 0001691D 0076DC00DC6666      <1>
398 00016924 66666600000076DC00- <1>     db  066h, 066h, 066h, 000h, 000h, 000h, 076h, 0dch, 000h, 0c6h, 0e6h, 0f6h, 0feh, 0deh, 0ceh, 0c6h
398 0001692D C6E6F6FEDECEC6      <1>
399 00016934 C6000000003C6C6C3E- <1>     db  0c6h, 000h, 000h, 000h, 000h, 03ch, 06ch, 06ch, 03eh, 000h, 07eh, 000h, 000h, 000h, 000h, 000h
399 0001693D 007E00000000000     <1>
400 00016944 000000386C6C38007C- <1>     db  000h, 000h, 000h, 038h, 06ch, 06ch, 038h, 000h, 07ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h
400 0001694D 00000000000000      <1>
401 00016954 0000303000303060C6- <1>     db  000h, 000h, 030h, 030h, 000h, 030h, 030h, 060h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h
401 0001695D C67C00000000000     <1>
402 00016964 00000000FEC0C0C000- <1>     db  000h, 000h, 000h, 000h, 0feh, 0c0h, 0c0h, 0c0h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
402 0001696D 00000000000000      <1>
403 00016974 0000FE0606006000000- <1>     db  000h, 000h, 0feh, 006h, 006h, 006h, 000h, 000h, 000h, 000h, 000h, 0c0h, 0c0h, 0c6h, 0cch, 0d8h
403 0001697D 0000C0C0C6CCD8      <1>
404 00016984 3060DC860C183E0000- <1>     db  030h, 060h, 0dch, 086h, 00ch, 018h, 03eh, 000h, 000h, 0c0h, 0c0h, 0c6h, 0cch, 0d8h, 030h, 066h
404 0001698D C0C0C6CCD83066      <1>
405 00016994 CE9E3E060600000018- <1>     db  0ceh, 09eh, 03eh, 006h, 006h, 000h, 000h, 000h, 018h, 018h, 000h, 018h, 018h, 03ch, 03ch, 03ch
405 0001699D 180018183C3C3C      <1>
406 000169A4 180000000000000036- <1>     db  018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 036h, 06ch, 0d8h, 06ch, 036h, 000h, 000h, 000h
406 000169AD 6CD86C36000000      <1>
407 000169B4 000000000000D86C36- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 0d8h, 06ch, 036h, 06ch, 0d8h, 000h, 000h, 000h, 000h, 000h
407 000169BD 6CD80000000000      <1>
408 000169C4 114411441144114411- <1>     db  011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 055h, 0aah
408 000169CD 441144114455AA      <1>
409 000169D4 55AA55AA55AA55AA55- <1>     db  055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah, 0ddh, 077h, 0ddh, 077h
409 000169DD AA55AADD77DD77      <1>
410 000169E4 DD77DD77DD77DD77DD- <1>     db  0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 018h, 018h, 018h, 018h, 018h, 018h
410 000169ED 77181818181818      <1>
411 000169F4 1818181818181818- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 0f8h
411 000169FD 181818181818F8      <1>
412 00016A04 1818181818181818- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 0f8h, 018h, 0f8h, 018h, 018h
412 00016A0D 1818F818F81818      <1>
413 00016A14 1818181836363636- <1>     db  018h, 018h, 018h, 018h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 0f6h, 036h, 036h, 036h, 036h
413 00016A1D 3636F636363636      <1>
414 00016A24 3636000000000000- <1>     db  036h, 036h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0feh, 036h, 036h, 036h, 036h, 036h, 036h
414 00016A2D FE363636363636      <1>
415 00016A34 0000000000F818F818- <1>     db  000h, 000h, 000h, 000h, 000h, 0f8h, 018h, 0f8h, 018h, 018h, 018h, 018h, 018h, 018h, 036h, 036h
415 00016A3D 18181818183636      <1>
416 00016A44 363636F606F6363636- <1>     db  036h, 036h, 036h, 0f6h, 006h, 0f6h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
416 00016A4D 36363636363636      <1>
417 00016A54 3636363636363636- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 000h, 000h, 000h, 000h, 000h, 0feh
417 00016A5D 3600000000000FE     <1>
418 00016A64 06F6363636363636- <1>     db  006h, 0f6h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 0f6h, 006h, 0feh
418 00016A6D 36363636F606FE      <1>
419 00016A74 0000000000000363636- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 0feh, 000h, 000h
419 00016A7D 36363636FE0000      <1>
420 00016A84 0000000001818181818- <1>     db  000h, 000h, 000h, 000h, 018h, 018h, 018h, 018h, 018h, 0f8h, 018h, 0f8h, 000h, 000h, 000h, 000h
420 00016A8D F818F800000000      <1>
421 00016A94 0000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0f8h, 018h, 018h, 018h, 018h, 018h, 018h
421 00016A9D F8181818181818      <1>
422 00016AA4 1818181818181F00- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 01fh, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h
422 00016AAD 00000000001818      <1>
423 00016AB4 1818181818FF000000- <1>     db  018h, 018h, 018h, 018h, 018h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
423 00016ABD 00000000000000      <1>
424 00016AC4 000000FF1818181818- <1>     db  000h, 000h, 000h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
424 00016ACD 18181818181818      <1>
425 00016AD4 181F18181818181800- <1>     db  018h, 01fh, 018h, 018h, 018h, 018h, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh
425 00016ADD 0000000000000FF     <1>
426 00016AE4 0000000000000181818- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 0ffh, 018h, 018h
426 00016AED 18181818FF1818      <1>
427 00016AF4 1818181818181818- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 01fh, 018h, 01fh, 018h, 018h, 018h, 018h
427 00016AFD 1F181F18181818      <1>
428 00016B04 1818363636363636- <1>     db  018h, 018h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 037h, 036h, 036h, 036h, 036h, 036h, 036h
428 00016B0D 37363636363636      <1>
429 00016B14 3636363636373037F00- <1>     db  036h, 036h, 036h, 036h, 036h, 037h, 030h, 03fh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
429 00016B1D 00000000000000      <1>
430 00016B24 0000003F3037363636- <1>     db  000h, 000h, 000h, 03fh, 030h, 037h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
430 00016B2D 36363636363636      <1>
431 00016B34 36F700FF0000000000- <1>     db  036h, 0f7h, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh
431 00016B3D 0000000000000FF     <1>
432 00016B44 00F7363636363636- <1>     db  000h, 0f7h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 037h, 030h, 037h
432 00016B4D 36363636373037      <1>
433 00016B54 3636363636360000000- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 000h, 000h, 000h, 000h, 000h, 0ffh, 000h, 0ffh, 000h, 000h
433 00016B5D 0000FF00FF0000      <1>
434 00016B64 0000000036363636- <1>     db  000h, 000h, 000h, 000h, 036h, 036h, 036h, 036h, 036h, 0f7h, 000h, 0f7h, 036h, 036h, 036h, 036h
434 00016B6D F700F736363636      <1>
435 00016B74 3636181818181FF00- <1>     db  036h, 036h, 018h, 018h, 018h, 018h, 018h, 0ffh, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h
435 00016B7D FF0000000000000     <1>
436 00016B84 3636363636363FF00- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
436 00016B8D 00000000000000      <1>
437 00016B94 000000FF00FF181818- <1>     db  000h, 000h, 000h, 0ffh, 000h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 000h, 000h, 000h, 000h
437 00016B9D 18181800000000      <1>
438 00016BA4 000000FF3636363636- <1>     db  000h, 000h, 000h, 0ffh, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
438 00016BAD 36363636363636      <1>
439 00016BB4 363F000000000000018- <1>     db  036h, 03fh, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 018h, 018h, 018h, 01fh, 018h, 01fh
439 00016BBD 181818181F181F      <1>
440 00016BC4 0000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 01fh, 018h, 01fh, 018h, 018h
440 00016BCD 00001F181F1818      <1>
441 00016BD4 1818181800000000000- <1>     db  018h, 018h, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 03fh, 036h, 036h, 036h, 036h
441 00016BDD 00003F36363636      <1>
442 00016BE4 3636363636363636- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 0ffh, 036h, 036h, 036h, 036h, 036h, 036h
442 00016BED FF363636363636      <1>
443 00016BF4 1818181818FF18FF18- <1>     db  018h, 018h, 018h, 018h, 018h, 0ffh, 018h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
443 00016BFD 18181818181818      <1>
444 00016C04 1818181818F8000000- <1>     db  018h, 018h, 018h, 018h, 018h, 0f8h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
444 00016C0D 00000000000000      <1>
445 00016C14 0000001F1818181818- <1>     db  000h, 000h, 000h, 01fh, 018h, 018h, 018h, 018h, 018h, 018h, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh
445 00016C1D 18FFFFFFFFFFFF      <1>
446 00016C24 FFFFFFFFFFFFFFFF00- <1>     db  0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh
446 00016C2D 0000000000000FF     <1>
447 00016C34 FFFFFFFFFFFFF0F0F0- <1>     db  0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h
447 00016C3D F0F0F0F0F0F0F0      <1>
448 00016C44 F0F0F0F00F0F0F0F0F- <1>     db  0f0h, 0f0h, 0f0h, 0f0h, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh
448 00016C4D 0F0F0F0F0F0F0F      <1>
449 00016C54 0F0FFFFFFFFFFFFFFF- <1>     db  00fh, 00fh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h
449 00016C5D 00000000000000      <1>
450 00016C64 0000000000076DCD8D8- <1>     db  000h, 000h, 000h, 000h, 000h, 076h, 0dch, 0d8h, 0d8h, 0dch, 076h, 000h, 000h, 000h, 000h, 000h
450 00016C6D DC760000000000      <1>
451 00016C74 00007CC6FCC6C6FCC0- <1>     db  000h, 000h, 07ch, 0c6h, 0fch, 0c6h, 0c6h, 0fch, 0c0h, 0c0h, 040h, 000h, 000h, 000h, 0feh, 0c6h
451 00016C7D C040000000FEC6      <1>
452 00016C84 C6C0C0C0C0C0C00000- <1>     db  0c6h, 0c0h, 0c0h, 0c0h, 0c0h, 0c0h, 0c0h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0feh, 06ch
452 00016C8D 0000000000FE6C      <1>
453 00016C94 6C6C6C6C6C00000000- <1>     db  06ch, 06ch, 06ch, 06ch, 06ch, 000h, 000h, 000h, 000h, 000h, 0feh, 0c6h, 060h, 030h, 018h, 030h
453 00016C9D 00FEC660301830      <1>
```

```
454 00016CA4 60C6FE000000000000- <1>     db  060h, 0c6h, 0feh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 07eh, 0d8h, 0d8h, 0d8h, 0d8h
454 00016CAD 00007ED8D8D8D8      <1>
455 00016CB4 7000000000000000066- <1>     db  070h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 066h, 066h, 066h, 066h, 07ch, 060h, 060h, 0c0h
455 00016CBD 6666667C6060C0      <1>
456 00016CC4 00000000000076DC18- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 076h, 0dch, 018h, 018h, 018h, 018h, 018h, 000h, 000h, 000h
456 00016CCD 18181818000000      <1>
457 00016CD4 00007E183C6666663C- <1>     db  000h, 000h, 07eh, 018h, 03ch, 066h, 066h, 066h, 03ch, 018h, 07eh, 000h, 000h, 000h, 000h, 000h
457 00016CDD 187E0000000000      <1>
458 00016CE4 386CC6C6FEC6C66C38- <1>     db  038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 06ch, 038h, 000h, 000h, 000h, 000h, 000h, 038h, 06ch
458 00016CED 0000000000386C      <1>
459 00016CF4 C6C6C66C6C6CEE0000- <1>     db  0c6h, 0c6h, 0c6h, 06ch, 06ch, 06ch, 0eeh, 000h, 000h, 000h, 000h, 000h, 01eh, 030h, 018h, 00ch
459 00016CFD 0000001E30180C      <1>
460 00016D04 3E6666663C00000000- <1>     db  03eh, 066h, 066h, 066h, 03ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 07eh, 0dbh, 0dbh
460 00016D0D 000000007EDBDB      <1>
461 00016D14 7E0000000000000003- <1>     db  07eh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 003h, 006h, 07eh, 0dbh, 0dbh, 0f3h, 07eh, 060h
461 00016D1D 067EDBDBF37E60      <1>
462 00016D24 C000000000001C3060- <1>     db  0c0h, 000h, 000h, 000h, 000h, 000h, 01ch, 030h, 060h, 060h, 07ch, 060h, 060h, 030h, 01ch, 000h
462 00016D2D 607C6060301C00      <1>
463 00016D34 00000000007CC6C6C6- <1>     db  000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h
463 00016D3D C6C6C6C6000000      <1>
464 00016D44 000000FE0000FE0000- <1>     db  000h, 000h, 000h, 0feh, 000h, 000h, 0feh, 000h, 000h, 0feh, 000h, 000h, 000h, 000h, 000h, 000h
464 00016D4D FE000000000000      <1>
465 00016D54 0018187E18180000FF- <1>     db  000h, 018h, 018h, 07eh, 018h, 018h, 000h, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 030h, 018h
465 00016D5D 00000000003018      <1>
466 00016D64 0C060C1830007E0000- <1>     db  00ch, 006h, 00ch, 018h, 030h, 000h, 07eh, 000h, 000h, 000h, 000h, 000h, 00ch, 018h, 030h, 060h
466 00016D6D 0000000C183060      <1>
467 00016D74 30180C007E00000000- <1>     db  030h, 018h, 00ch, 000h, 07eh, 000h, 000h, 000h, 000h, 000h, 00eh, 01bh, 01bh, 018h, 018h, 018h
467 00016D7D 000E1B1B181818      <1>
468 00016D84 1818181818181818- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 0d8h, 0d8h
468 00016D8D 1818181818D8D8      <1>
469 00016D94 7000000000000001818- <1>     db  070h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 000h, 07eh, 000h, 018h, 018h, 000h, 000h
469 00016D9D 007E0018180000      <1>
470 00016DA4 00000000000076DC00- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 076h, 0dch, 000h, 076h, 0dch, 000h, 000h, 000h, 000h, 000h
470 00016DAD 76DC0000000000      <1>
471 00016DB4 00386C6C3800000000- <1>     db  000h, 038h, 06ch, 06ch, 038h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
471 00016DBD 00000000000000      <1>
472 00016DC4 0000000001818000000- <1>     db  000h, 000h, 000h, 000h, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
472 00016DCD 00000000000000      <1>
473 00016DD4 000000180000000000- <1>     db  000h, 000h, 000h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 00fh, 00ch, 00ch, 00ch, 00ch
473 00016DDD 00000F0C0C0C0C      <1>
474 00016DE4 0CEC6C3C1C00000000- <1>     db  00ch, 0ech, 06ch, 03ch, 01ch, 000h, 000h, 000h, 000h, 0d8h, 06ch, 06ch, 06ch, 06ch, 06ch, 000h
474 00016DED D86C6C6C6C6C00      <1>
475 00016DF4 0000000000000070D8- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 070h, 0d8h, 030h, 060h, 0c8h, 0f8h, 000h, 000h, 000h
475 00016DFD 3060C8F8000000      <1>
476 00016E04 00000000000000007C- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 07ch, 07ch, 07ch, 07ch, 07ch, 07ch, 000h, 000h
476 00016E0D 7C7C7C7C7C0000      <1>
477 00016E14 000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
477 00016E1D 00000000000000      <1>
478                              <1> vgafont16:
479 00016E24 000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
479 00016E2D 00000000000000      <1>
480 00016E34 00007E81A58181BD99- <1>     db  000h, 000h, 07eh, 081h, 0a5h, 081h, 081h, 0bdh, 099h, 081h, 081h, 07eh, 000h, 000h, 000h, 000h
480 00016E3D 81817E00000000      <1>
481 00016E44 00007EFFDBFFFFC3E7- <1>     db  000h, 000h, 07eh, 0ffh, 0dbh, 0ffh, 0ffh, 0c3h, 0e7h, 0ffh, 0ffh, 07eh, 000h, 000h, 000h, 000h
481 00016E4D FFFF7E00000000      <1>
482 00016E54 000000006CFEFEFEFE- <1>     db  000h, 000h, 000h, 000h, 06ch, 0feh, 0feh, 0feh, 0feh, 07ch, 038h, 010h, 000h, 000h, 000h, 000h
482 00016E5D 7C381000000000      <1>
483 00016E64 0000000010387CFE7C- <1>     db  000h, 000h, 000h, 000h, 010h, 038h, 07ch, 0feh, 07ch, 038h, 010h, 000h, 000h, 000h, 000h, 000h
483 00016E6D 38100000000000      <1>
484 00016E74 000000183C3CE7E7E7- <1>     db  000h, 000h, 000h, 018h, 03ch, 03ch, 0e7h, 0e7h, 0e7h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h
484 00016E7D 18183C00000000      <1>
485 00016E84 000000183C7EFFFF7E- <1>     db  000h, 000h, 000h, 018h, 03ch, 07eh, 0ffh, 0ffh, 07eh, 018h, 018h, 03ch, 000h, 000h, 000h, 000h
485 00016E8D 18183C00000000      <1>
486 00016E94 000000000000183C3C- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 018h, 03ch, 03ch, 018h, 000h, 000h, 000h, 000h, 000h, 000h
486 00016E9D 18000000000000      <1>
487 00016EA4 FFFFFFFFFFFFE7C3C3- <1>     db  0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0e7h, 0c3h, 0c3h, 0e7h, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh
487 00016EAD E7FFFFFFFFFFFF      <1>
488 00016EB4 00000000003C664242- <1>     db  000h, 000h, 000h, 000h, 000h, 03ch, 066h, 042h, 042h, 066h, 03ch, 000h, 000h, 000h, 000h, 000h
488 00016EBD 663C0000000000      <1>
489 00016EC4 FFFFFFFFFFC399BDBD- <1>     db  0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0c3h, 099h, 0bdh, 0bdh, 099h, 0c3h, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh
489 00016ECD 99C3FFFFFFFFFF      <1>
490 00016ED4 00001E0E1A3278CCCC- <1>     db  000h, 000h, 01eh, 00eh, 01ah, 032h, 078h, 0cch, 0cch, 0cch, 0cch, 078h, 000h, 000h, 000h, 000h
490 00016EDD CCCC7800000000      <1>
491 00016EE4 00003C666666663C18- <1>     db  000h, 000h, 03ch, 066h, 066h, 066h, 066h, 03ch, 018h, 07eh, 018h, 018h, 000h, 000h, 000h, 000h
491 00016EED 7E181800000000      <1>
492 00016EF4 00003F333F30303030- <1>     db  000h, 000h, 03fh, 033h, 03fh, 030h, 030h, 030h, 030h, 070h, 0f0h, 0e0h, 000h, 000h, 000h, 000h
492 00016EFD 70F0E000000000      <1>
493 00016F04 00007F637F63636363- <1>     db  000h, 000h, 07fh, 063h, 07fh, 063h, 063h, 063h, 063h, 067h, 0e7h, 0e6h, 0c0h, 000h, 000h, 000h
493 00016F0D 67E7E6C0000000      <1>
494 00016F14 0000001818DB3CE73C- <1>     db  000h, 000h, 000h, 018h, 018h, 0dbh, 03ch, 0e7h, 03ch, 0dbh, 018h, 018h, 000h, 000h, 000h, 000h
494 00016F1D DB1818000000000     <1>
495 00016F24 0080C0E0F0F8FEF8F0- <1>     db  000h, 080h, 0c0h, 0e0h, 0f0h, 0f8h, 0feh, 0f8h, 0f0h, 0e0h, 0c0h, 080h, 000h, 000h, 000h, 000h
495 00016F2D E0C080000000000     <1>
496 00016F34 0002060E1E3EFE3E1E- <1>     db  000h, 002h, 006h, 00eh, 01eh, 03eh, 0feh, 03eh, 01eh, 00eh, 006h, 002h, 000h, 000h, 000h, 000h
496 00016F3D 0E060200000000      <1>
497 00016F44 0000183C7E1818187E- <1>     db  000h, 000h, 018h, 03ch, 07eh, 018h, 018h, 018h, 07eh, 03ch, 018h, 000h, 000h, 000h, 000h, 000h
497 00016F4D 3C180000000000      <1>
498 00016F54 000066666666666666- <1>     db  000h, 000h, 066h, 066h, 066h, 066h, 066h, 066h, 066h, 000h, 066h, 066h, 000h, 000h, 000h, 000h
498 00016F5D 00666600000000      <1>
499 00016F64 00007FDBDBDB7B1B1B- <1>     db  000h, 000h, 07fh, 0dbh, 0dbh, 0dbh, 07bh, 01bh, 01bh, 01bh, 01bh, 01bh, 000h, 000h, 000h, 000h
499 00016F6D 1B1B1B00000000      <1>
500 00016F74 007CC660386CC6C66C- <1>     db  000h, 07ch, 0c6h, 060h, 038h, 06ch, 0c6h, 0c6h, 06ch, 038h, 00ch, 0c6h, 07ch, 000h, 000h, 000h
500 00016F7D 380CC67C000000      <1>
501 00016F84 00000000000000000FE- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0feh, 0feh, 0feh, 0feh, 000h, 000h, 000h, 000h
501 00016F8D FEFEFE00000000      <1>
502 00016F94 0000183C7E1818187E- <1>     db  000h, 000h, 018h, 03ch, 07eh, 018h, 018h, 018h, 07eh, 03ch, 018h, 07eh, 000h, 000h, 000h, 000h
502 00016F9D 3C187E00000000      <1>
503 00016FA4 0000183C7E18181818- <1>     db  000h, 000h, 018h, 03ch, 07eh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 000h, 000h, 000h, 000h
503 00016FAD 18181800000000      <1>
504 00016FB4 000018181818181818- <1>     db  000h, 000h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 07eh, 03ch, 018h, 000h, 000h, 000h, 000h
504 00016FBD 7E3C1800000000      <1>
505 00016FC4 0000000000180CFE0C- <1>     db  000h, 000h, 000h, 000h, 000h, 018h, 00ch, 0feh, 00ch, 018h, 000h, 000h, 000h, 000h, 000h, 000h
505 00016FCD 18000000000000      <1>
506 00016FD4 00000000003060FE60- <1>     db  000h, 000h, 000h, 000h, 000h, 030h, 060h, 0feh, 060h, 030h, 000h, 000h, 000h, 000h, 000h, 000h
506 00016FDD 30000000000000      <1>
507 00016FE4 000000000000C0C0C0- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 0c0h, 0c0h, 0c0h, 0feh, 000h, 000h, 000h, 000h, 000h, 000h
507 00016FED FE0000000000      <1>
508 00016FF4 00000000002466FF66- <1>     db  000h, 000h, 000h, 000h, 000h, 024h, 066h, 0ffh, 066h, 024h, 000h, 000h, 000h, 000h, 000h, 000h
508 00016FFD 24000000000000      <1>
509 00017004 0000000001038387C7C- <1>     db  000h, 000h, 000h, 000h, 010h, 038h, 038h, 07ch, 07ch, 0feh, 0feh, 000h, 000h, 000h, 000h, 000h
509 0001700D FEFE0000000000      <1>
510 00017014 00000000FEFE7C7C38- <1>     db  000h, 000h, 000h, 000h, 0feh, 0feh, 07ch, 07ch, 038h, 038h, 010h, 000h, 000h, 000h, 000h, 000h
510 0001701D 38100000000000      <1>
511 00017024 000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
511 0001702D 00000000000000      <1>
512 00017034 0000183C3C3C181818- <1>     db  000h, 000h, 018h, 03ch, 03ch, 03ch, 018h, 018h, 018h, 000h, 018h, 018h, 000h, 000h, 000h, 000h
512 0001703D 00181800000000      <1>
513 00017044 006666662400000000- <1>     db  000h, 066h, 066h, 066h, 024h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
```

```
513 0001704D 00000000000000      <1>
514 00017054 0000006C6CFE6C6C6C- <1>     db  000h, 000h, 000h, 06ch, 06ch, 0feh, 06ch, 06ch, 06ch, 0feh, 06ch, 06ch, 000h, 000h, 000h, 000h
514 0001705D FE6C6C00000000      <1>
515 00017064 18187CC6C2C07C0606- <1>     db  018h, 018h, 07ch, 0c6h, 0c2h, 0c0h, 07ch, 006h, 006h, 086h, 0c6h, 07ch, 018h, 018h, 000h, 000h
515 0001706D 86C67C18180000      <1>
516 00017074 00000000C2C60C1830- <1>     db  000h, 000h, 000h, 000h, 0c2h, 0c6h, 00ch, 018h, 030h, 060h, 0c6h, 086h, 000h, 000h, 000h, 000h
516 0001707D 60C68600000000      <1>
517 00017084 0000386C6C3876DCCC- <1>     db  000h, 000h, 038h, 06ch, 06ch, 038h, 076h, 0dch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
517 0001708D CCCC7600000000      <1>
518 00017094 003030306000000000- <1>     db  000h, 030h, 030h, 030h, 060h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
518 0001709D 00000000000000      <1>
519 000170A4 00000C183030303030- <1>     db  000h, 000h, 00ch, 018h, 030h, 030h, 030h, 030h, 030h, 030h, 018h, 00ch, 000h, 000h, 000h, 000h
519 000170AD 30180C00000000      <1>
520 000170B4 000030180C0C0C0C0C- <1>     db  000h, 000h, 030h, 018h, 00ch, 00ch, 00ch, 00ch, 00ch, 00ch, 018h, 030h, 000h, 000h, 000h, 000h
520 000170BD 0C183000000000      <1>
521 000170C4 0000000000663CFF3C- <1>     db  000h, 000h, 000h, 000h, 000h, 066h, 03ch, 0ffh, 03ch, 066h, 000h, 000h, 000h, 000h, 000h, 000h
521 000170CD 66000000000000      <1>
522 000170D4 000000000018187E18- <1>     db  000h, 000h, 000h, 000h, 000h, 018h, 018h, 07eh, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 000h
522 000170DD 18000000000000      <1>
523 000170E4 00000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 018h, 030h, 000h, 000h, 000h
523 000170ED 18181830000000      <1>
524 000170F4 00000000000000FE00- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0feh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
524 000170FD 00000000000000      <1>
525 00017104 00000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 000h, 000h, 000h, 000h
525 0001710D 00181800000000      <1>
526 00017114 0000000002060C1830- <1>     db  000h, 000h, 000h, 000h, 002h, 006h, 00ch, 018h, 030h, 060h, 0c0h, 080h, 000h, 000h, 000h, 000h
526 0001711D 60C08000000000      <1>
527 00017124 00003C66C3C3DBDBC3- <1>     db  000h, 000h, 03ch, 066h, 0c3h, 0c3h, 0dbh, 0dbh, 0c3h, 0c3h, 066h, 03ch, 000h, 000h, 000h, 000h
527 0001712D C3663C00000000      <1>
528 00017134 000018387818181818- <1>     db  000h, 000h, 018h, 038h, 078h, 018h, 018h, 018h, 018h, 018h, 018h, 07eh, 000h, 000h, 000h, 000h
528 0001713D 18187E00000000      <1>
529 00017144 00007CC6060C183060- <1>     db  000h, 000h, 07ch, 0c6h, 006h, 00ch, 018h, 030h, 060h, 0c0h, 0c6h, 0feh, 000h, 000h, 000h, 000h
529 0001714D C0C6FE00000000      <1>
530 00017154 00007CC606063C0606- <1>     db  000h, 000h, 07ch, 0c6h, 006h, 006h, 03ch, 006h, 006h, 006h, 0c6h, 07ch, 000h, 000h, 000h, 000h
530 0001715D 06C67C00000000      <1>
531 00017164 00000C1C3C6CCCFE0C- <1>     db  000h, 000h, 00ch, 01ch, 03ch, 06ch, 0cch, 0feh, 00ch, 00ch, 00ch, 01eh, 000h, 000h, 000h, 000h
531 0001716D 0C0C1E00000000      <1>
532 00017174 0000FEC0C0C0FC0606- <1>     db  000h, 000h, 0feh, 0c0h, 0c0h, 0c0h, 0fch, 006h, 006h, 006h, 0c6h, 07ch, 000h, 000h, 000h, 000h
532 0001717D 06C67C00000000      <1>
533 00017184 00003860C0C0FCC6C6- <1>     db  000h, 000h, 038h, 060h, 0c0h, 0c0h, 0fch, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
533 0001718D C6C67C00000000      <1>
534 00017194 0000FEC606060C1830- <1>     db  000h, 000h, 0feh, 0c6h, 006h, 006h, 00ch, 018h, 030h, 030h, 030h, 030h, 000h, 000h, 000h, 000h
534 0001719D 30303000000000      <1>
535 000171A4 00007CC6C6C67CC6C6- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
535 000171AD C6C67C00000000      <1>
536 000171B4 00007CC6C6C67E0606- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 07eh, 006h, 006h, 006h, 00ch, 078h, 000h, 000h, 000h, 000h
536 000171BD 060C7800000000      <1>
537 000171C4 000000001818000000- <1>     db  000h, 000h, 000h, 000h, 018h, 018h, 000h, 000h, 000h, 018h, 018h, 000h, 000h, 000h, 000h, 000h
537 000171CD 18180000000000      <1>
538 000171D4 000000001818000000- <1>     db  000h, 000h, 000h, 000h, 018h, 018h, 000h, 000h, 000h, 018h, 018h, 030h, 000h, 000h, 000h, 000h
538 000171DD 18183000000000      <1>
539 000171E4 000000060C18306030- <1>     db  000h, 000h, 000h, 006h, 00ch, 018h, 030h, 060h, 030h, 018h, 00ch, 006h, 000h, 000h, 000h, 000h
539 000171ED 180C0600000000      <1>
540 000171F4 00000000007E00007E- <1>     db  000h, 000h, 000h, 000h, 000h, 07eh, 000h, 000h, 07eh, 000h, 000h, 000h, 000h, 000h, 000h, 000h
540 000171FD 00000000000000      <1>
541 00017204 0000006030180C060C- <1>     db  000h, 000h, 000h, 060h, 030h, 018h, 00ch, 006h, 00ch, 018h, 030h, 060h, 000h, 000h, 000h, 000h
541 0001720D 18306000000000      <1>
542 00017214 00007CC6C60C181818- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 00ch, 018h, 018h, 018h, 000h, 018h, 018h, 000h, 000h, 000h, 000h
542 0001721D 00181800000000      <1>
543 00017224 0000007CC6C6DEDEDE- <1>     db  000h, 000h, 000h, 07ch, 0c6h, 0c6h, 0deh, 0deh, 0deh, 0dch, 0c0h, 07ch, 000h, 000h, 000h, 000h
543 0001722D DCC07C00000000      <1>
544 00017234 000010386CC6C6FEC6- <1>     db  000h, 000h, 010h, 038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h, 000h
544 0001723D C6C6C600000000      <1>
545 00017244 0000FC6666667C6666- <1>     db  000h, 000h, 0fch, 066h, 066h, 066h, 07ch, 066h, 066h, 066h, 066h, 0fch, 000h, 000h, 000h, 000h
545 0001724D 6666FC00000000      <1>
546 00017254 00003C66C2C0C0C0C0- <1>     db  000h, 000h, 03ch, 066h, 0c2h, 0c0h, 0c0h, 0c0h, 0c0h, 0c2h, 066h, 03ch, 000h, 000h, 000h, 000h
546 0001725D C2663C00000000      <1>
547 00017264 0000F86C6666666666- <1>     db  000h, 000h, 0f8h, 06ch, 066h, 066h, 066h, 066h, 066h, 066h, 06ch, 0f8h, 000h, 000h, 000h, 000h
547 0001726D 666CF800000000      <1>
548 00017274 0000FE666268786860- <1>     db  000h, 000h, 0feh, 066h, 062h, 068h, 078h, 068h, 060h, 062h, 066h, 0feh, 000h, 000h, 000h, 000h
548 0001727D 6266FE00000000      <1>
549 00017284 0000FE666268786860- <1>     db  000h, 000h, 0feh, 066h, 062h, 068h, 078h, 068h, 060h, 060h, 060h, 0f0h, 000h, 000h, 000h, 000h
549 0001728D 6060F000000000      <1>
550 00017294 00003C66C2C0C0DEC6- <1>     db  000h, 000h, 03ch, 066h, 0c2h, 0c0h, 0c0h, 0deh, 0c6h, 0c6h, 066h, 03ah, 000h, 000h, 000h, 000h
550 0001729D C6663A00000000      <1>
551 000172A4 0000C6C6C6C6FEC6C6- <1>     db  000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h, 000h
551 000172AD C6C6C600000000      <1>
552 000172B4 00003C181818181818- <1>     db  000h, 000h, 03ch, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h
552 000172BD 18183C00000000      <1>
553 000172C4 00001E0C0C0C0C0CCC- <1>     db  000h, 000h, 01eh, 00ch, 00ch, 00ch, 00ch, 00ch, 0cch, 0cch, 0cch, 078h, 000h, 000h, 000h, 000h
553 000172CD CCCC7800000000      <1>
554 000172D4 0000E666666C78786C- <1>     db  000h, 000h, 0e6h, 066h, 066h, 06ch, 078h, 078h, 06ch, 066h, 066h, 0e6h, 000h, 000h, 000h, 000h
554 000172DD 6666E600000000      <1>
555 000172E4 0000F0606060606060- <1>     db  000h, 000h, 0f0h, 060h, 060h, 060h, 060h, 060h, 060h, 062h, 066h, 0feh, 000h, 000h, 000h, 000h
555 000172ED 6266FE00000000      <1>
556 000172F4 0000C3E7FFFFDBC3C3- <1>     db  000h, 000h, 0c3h, 0e7h, 0ffh, 0ffh, 0dbh, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 000h, 000h, 000h, 000h
556 000172FD C3C3C300000000      <1>
557 00017304 0000C6E6F6FEDECEC6- <1>     db  000h, 000h, 0c6h, 0e6h, 0f6h, 0feh, 0deh, 0ceh, 0c6h, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h, 000h
557 0001730D C6C6C600000000      <1>
558 00017314 00007CC6C6C6C6C6C6- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
558 0001731D C6C67C00000000      <1>
559 00017324 0000FC6666667C6060- <1>     db  000h, 000h, 0fch, 066h, 066h, 066h, 07ch, 060h, 060h, 060h, 060h, 0f0h, 000h, 000h, 000h, 000h
559 0001732D 6060F000000000      <1>
560 00017334 00007CC6C6C6C6C6C6- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0d6h, 0deh, 07ch, 00ch, 00eh, 000h, 000h
560 0001733D D6DE7C0C0E0000      <1>
561 00017344 0000FC6666667C6C66- <1>     db  000h, 000h, 0fch, 066h, 066h, 066h, 07ch, 06ch, 066h, 066h, 066h, 0e6h, 000h, 000h, 000h, 000h
561 0001734D 6666E600000000      <1>
562 00017354 00007CC6C660380C06- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 060h, 038h, 00ch, 006h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
562 0001735D C6C67C00000000      <1>
563 00017364 0000FFDB9918181818- <1>     db  000h, 000h, 0ffh, 0dbh, 099h, 018h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h
563 0001736D 18183C00000000      <1>
564 00017374 0000C6C6C6C6C6C6C6- <1>     db  000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
564 0001737D C6C67C00000000      <1>
565 00017384 0000C3C3C3C3C3C3C3- <1>     db  000h, 000h, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 066h, 03ch, 018h, 000h, 000h, 000h, 000h
565 0001738D 663C1800000000      <1>
566 00017394 0000C3C3C3C3C3DBDB- <1>     db  000h, 000h, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 0dbh, 0dbh, 0ffh, 066h, 066h, 000h, 000h, 000h, 000h
566 0001739D FF666600000000      <1>
567 000173A4 0000C3C3663C18183C- <1>     db  000h, 000h, 0c3h, 0c3h, 066h, 03ch, 018h, 018h, 03ch, 066h, 0c3h, 0c3h, 000h, 000h, 000h, 000h
567 000173AD 66C3C300000000      <1>
568 000173B4 0000C3C3C3663C1818- <1>     db  000h, 000h, 0c3h, 0c3h, 0c3h, 066h, 03ch, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h
568 000173BD 18183C00000000      <1>
569 000173C4 0000FFC3860C183060- <1>     db  000h, 000h, 0ffh, 0c3h, 086h, 00ch, 018h, 030h, 060h, 0c1h, 0c3h, 0ffh, 000h, 000h, 000h, 000h
569 000173CD C1C3FF00000000      <1>
570 000173D4 00003C303030303030- <1>     db  000h, 000h, 03ch, 030h, 030h, 030h, 030h, 030h, 030h, 030h, 030h, 03ch, 000h, 000h, 000h, 000h
570 000173DD 30303C00000000      <1>
571 000173E4 00000080C0E070381C- <1>     db  000h, 000h, 000h, 080h, 0c0h, 0e0h, 070h, 038h, 01ch, 00eh, 006h, 002h, 000h, 000h, 000h, 000h
571 000173ED 0E060200000000      <1>
572 000173F4 00003C0C0C0C0C0C0C- <1>     db  000h, 000h, 03ch, 00ch, 00ch, 00ch, 00ch, 00ch, 00ch, 00ch, 00ch, 03ch, 000h, 000h, 000h, 000h
```

```
572 000173FD 0C0C3C00000000      <1>
573 00017404 10386CC60000000000- <1>     db  010h, 038h, 06ch, 0c6h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
573 0001740D 00000000000000      <1>
574 00017414 00000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh, 000h, 000h
574 0001741D 00000000FF0000      <1>
575 00017424 3030180000000000000- <1>     db  030h, 030h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
575 0001742D 00000000000000      <1>
576 00017434 0000000000780C7CCC- <1>     db  000h, 000h, 000h, 000h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
576 0001743D CCCC7600000000      <1>
577 00017444 0000E06060786C6666- <1>     db  000h, 000h, 0e0h, 060h, 060h, 078h, 06ch, 066h, 066h, 066h, 066h, 07ch, 000h, 000h, 000h, 000h
577 0001744D 66667C00000000      <1>
578 00017454 00000000007CC6C0C0- <1>     db  000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 0c0h, 0c0h, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h
578 0001745D C0C67C00000000      <1>
579 00017464 00001C0C0C3C6CCCCC- <1>     db  000h, 000h, 01ch, 00ch, 00ch, 03ch, 06ch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
579 0001746D CCCC7600000000      <1>
580 00017474 00000000007CC6FEC0- <1>     db  000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 0feh, 0c0h, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h
580 0001747D C0C67C00000000      <1>
581 00017484 0000386C6460F06060- <1>     db  000h, 000h, 038h, 06ch, 064h, 060h, 0f0h, 060h, 060h, 060h, 060h, 0f0h, 000h, 000h, 000h, 000h
581 0001748D 6060F000000000      <1>
582 00017494 000000000076CCCCCC- <1>     db  000h, 000h, 000h, 000h, 000h, 076h, 0cch, 0cch, 0cch, 0cch, 0cch, 07ch, 00ch, 0cch, 078h, 000h
582 0001749D CCCC7C0CCC7800      <1>
583 000174A4 0000E060606C766666- <1>     db  000h, 000h, 0e0h, 060h, 060h, 06ch, 076h, 066h, 066h, 066h, 066h, 0e6h, 000h, 000h, 000h, 000h
583 000174AD 6666E600000000      <1>
584 000174B4 000018180038181818- <1>     db  000h, 000h, 018h, 018h, 000h, 038h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h
584 000174BD 18183C00000000      <1>
585 000174C4 00000606000E060606- <1>     db  000h, 000h, 006h, 006h, 000h, 00eh, 006h, 006h, 006h, 006h, 006h, 006h, 066h, 066h, 03ch, 000h
585 000174CD 06060666663C00      <1>
586 000174D4 0000E06060666C7878- <1>     db  000h, 000h, 0e0h, 060h, 060h, 066h, 06ch, 078h, 078h, 06ch, 066h, 0e6h, 000h, 000h, 000h, 000h
586 000174DD 6C66E600000000      <1>
587 000174E4 000038181818181818- <1>     db  000h, 000h, 038h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h
587 000174ED 18183C00000000      <1>
588 000174F4 0000000000E6FFDBDB- <1>     db  000h, 000h, 000h, 000h, 000h, 0e6h, 0ffh, 0dbh, 0dbh, 0dbh, 0dbh, 0dbh, 000h, 000h, 000h, 000h
588 000174FD DBDBDB00000000      <1>
589 00017504 0000000000DC666666- <1>     db  000h, 000h, 000h, 000h, 000h, 0dch, 066h, 066h, 066h, 066h, 066h, 066h, 000h, 000h, 000h, 000h
589 0001750D 66666600000000      <1>
590 00017514 00000000007CC6C6C6- <1>     db  000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
590 0001751D C6C67C00000000      <1>
591 00017524 0000000000DC666666- <1>     db  000h, 000h, 000h, 000h, 000h, 0dch, 066h, 066h, 066h, 066h, 066h, 07ch, 060h, 060h, 0f0h, 000h
591 0001752D 66667C6060F000      <1>
592 00017534 000000000076CCCCCC- <1>     db  000h, 000h, 000h, 000h, 000h, 076h, 0cch, 0cch, 0cch, 0cch, 0cch, 07ch, 00ch, 00ch, 01eh, 000h
592 0001753D CCCC7C0C0C1E00      <1>
593 00017544 0000000000DC766660- <1>     db  000h, 000h, 000h, 000h, 000h, 0dch, 076h, 066h, 060h, 060h, 060h, 0f0h, 000h, 000h, 000h, 000h
593 0001754D 6060F000000000      <1>
594 00017554 00000000007CC66038- <1>     db  000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 060h, 038h, 00ch, 0c6h, 07ch, 000h, 000h, 000h, 000h
594 0001755D 0CC67C00000000      <1>
595 00017564 0000103030FC303030- <1>     db  000h, 000h, 010h, 030h, 030h, 0fch, 030h, 030h, 030h, 030h, 036h, 01ch, 000h, 000h, 000h, 000h
595 0001756D 30361C00000000      <1>
596 00017574 0000000000CCCCCCCC- <1>     db  000h, 000h, 000h, 000h, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
596 0001757D CCCC7600000000      <1>
597 00017584 0000000000C3C3C3C3- <1>     db  000h, 000h, 000h, 000h, 000h, 0c3h, 0c3h, 0c3h, 0c3h, 066h, 03ch, 018h, 000h, 000h, 000h, 000h
597 0001758D 663C1800000000      <1>
598 00017594 0000000000C3C3C3DB- <1>     db  000h, 000h, 000h, 000h, 000h, 0c3h, 0c3h, 0c3h, 0dbh, 0dbh, 0ffh, 066h, 000h, 000h, 000h, 000h
598 0001759D DBFF6600000000      <1>
599 000175A4 0000000000C3663C18- <1>     db  000h, 000h, 000h, 000h, 000h, 0c3h, 066h, 03ch, 018h, 03ch, 066h, 0c3h, 000h, 000h, 000h, 000h
599 000175AD 3C66C300000000      <1>
600 000175B4 0000000000C6C6C6C6- <1>     db  000h, 000h, 000h, 000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07eh, 006h, 00ch, 0f8h, 000h
600 000175BD C6C67E060CF800      <1>
601 000175C4 0000000000FECC1830- <1>     db  000h, 000h, 000h, 000h, 000h, 0feh, 0cch, 018h, 030h, 060h, 0c6h, 0feh, 000h, 000h, 000h, 000h
601 000175CD 60C6FE00000000      <1>
602 000175D4 00000E181818701818- <1>     db  000h, 000h, 00eh, 018h, 018h, 018h, 070h, 018h, 018h, 018h, 018h, 00eh, 000h, 000h, 000h, 000h
602 000175DD 18180E00000000      <1>
603 000175E4 0000181818001818- <1>     db  000h, 000h, 018h, 018h, 018h, 018h, 000h, 018h, 018h, 018h, 018h, 018h, 000h, 000h, 000h, 000h
603 000175ED 18181800000000      <1>
604 000175F4 0000701818180E1818- <1>     db  000h, 000h, 070h, 018h, 018h, 018h, 00eh, 018h, 018h, 018h, 018h, 070h, 000h, 000h, 000h, 000h
604 000175FD 18187000000000      <1>
605 00017604 000076DC0000000000- <1>     db  000h, 000h, 076h, 0dch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
605 0001760D 00000000000000      <1>
606 00017614 0000000010386CC6C6- <1>     db  000h, 000h, 000h, 000h, 010h, 038h, 06ch, 0c6h, 0c6h, 0c6h, 0feh, 000h, 000h, 000h, 000h, 000h
606 0001761D C6FE0000000000      <1>
607 00017624 00003C66C2C0C0C0C2- <1>     db  000h, 000h, 03ch, 066h, 0c2h, 0c0h, 0c0h, 0c0h, 0c2h, 066h, 03ch, 00ch, 006h, 07ch, 000h, 000h
607 0001762D 663C0C067C0000      <1>
608 00017634 0000CC0000CCCCCCCC- <1>     db  000h, 000h, 0cch, 000h, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
608 0001763D CCCC7600000000      <1>
609 00017644 000C1830007CC6FEC0- <1>     db  000h, 00ch, 018h, 030h, 000h, 07ch, 0c6h, 0feh, 0c0h, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h
609 0001764D C0C67C00000000      <1>
610 00017654 0010386C00780C7CCC- <1>     db  000h, 010h, 038h, 06ch, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
610 0001765D CCCC7600000000      <1>
611 00017664 0000CC0000780C7CCC- <1>     db  000h, 000h, 0cch, 000h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
611 0001766D CCCC7600000000      <1>
612 00017674 0060301800780C7CCC- <1>     db  000h, 060h, 030h, 018h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
612 0001767D CCCC7600000000      <1>
613 00017684 00386C3800780C7CCC- <1>     db  000h, 038h, 06ch, 038h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
613 0001768D CCCC7600000000      <1>
614 00017694 000000003C66606066- <1>     db  000h, 000h, 000h, 000h, 03ch, 066h, 060h, 060h, 066h, 03ch, 00ch, 006h, 03ch, 000h, 000h, 000h
614 0001769D 3C0C063C000000      <1>
615 000176A4 0010386C007CC6FEC0- <1>     db  000h, 010h, 038h, 06ch, 000h, 07ch, 0c6h, 0feh, 0c0h, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h
615 000176AD C0C67C00000000      <1>
616 000176B4 0000C600007CC6FEC0- <1>     db  000h, 000h, 0c6h, 000h, 000h, 07ch, 0c6h, 0feh, 0c0h, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h
616 000176BD C0C67C00000000      <1>
617 000176C4 00603018007CC6FEC0- <1>     db  000h, 060h, 030h, 018h, 000h, 07ch, 0c6h, 0feh, 0c0h, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h
617 000176CD C0C67C00000000      <1>
618 000176D4 000066000038181818- <1>     db  000h, 000h, 066h, 000h, 000h, 038h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h
618 000176DD 18183C00000000      <1>
619 000176E4 00183C660038181818- <1>     db  000h, 018h, 03ch, 066h, 000h, 038h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h
619 000176ED 18183C00000000      <1>
620 000176F4 0060301800381818 18- <1>     db  000h, 060h, 030h, 018h, 000h, 038h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h
620 000176FD 18183C00000000      <1>
621 00017704 00C60010386CC6C6FE- <1>     db  000h, 0c6h, 000h, 010h, 038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h, 000h
621 0001770D C6C6C600000000      <1>
622 00017714 386C3800386CC6C6FE- <1>     db  038h, 06ch, 038h, 000h, 038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h, 000h
622 0001771D C6C6C600000000      <1>
623 00017724 18306000FE66607C60- <1>     db  018h, 030h, 060h, 000h, 0feh, 066h, 060h, 07ch, 060h, 060h, 066h, 0feh, 000h, 000h, 000h, 000h
623 0001772D 6066FE00000000      <1>
624 00017734 00000000006E3B1B7E- <1>     db  000h, 000h, 000h, 000h, 000h, 06eh, 03bh, 01bh, 07eh, 0d8h, 0dch, 077h, 000h, 000h, 000h, 000h
624 0001773D D8DC7700000000      <1>
625 00017744 00003E6CCCCCFECCCC- <1>     db  000h, 000h, 03eh, 06ch, 0cch, 0cch, 0feh, 0cch, 0cch, 0cch, 0cch, 0ceh, 000h, 000h, 000h, 000h
625 0001774D CCCCCE00000000      <1>
626 00017754 0010386C007CC6C6C6- <1>     db  000h, 010h, 038h, 06ch, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
626 0001775D C6C67C00000000      <1>
627 00017764 0000C600007CC6C6C6- <1>     db  000h, 000h, 0c6h, 000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
627 0001776D C6C67C00000000      <1>
628 00017774 00603018007CC6C6C6- <1>     db  000h, 060h, 030h, 018h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
628 0001777D C6C67C00000000      <1>
629 00017784 003078CC00CCCCCCCC- <1>     db  000h, 030h, 078h, 0cch, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
629 0001778D CCCC7600000000      <1>
630 00017794 0060301800CCCCCCCC- <1>     db  000h, 060h, 030h, 018h, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
630 0001779D CCCC7600000000      <1>
631 000177A4 0000C60000C6C6C6C6- <1>     db  000h, 000h, 0c6h, 000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07eh, 006h, 00ch, 078h, 000h
```

```
631 000177AD C6C67E060C7800      <1>
632 000177B4 00C6007CC6C6C6C6C6- <1>     db  000h, 0c6h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
632 000177BD C6C67C00000000      <1>
633 000177C4 00C600C6C6C6C6C6C6- <1>     db  000h, 0c6h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
633 000177CD C6C67C00000000      <1>
634 000177D4 0018187EC3C0C0C0C3- <1>     db  000h, 018h, 018h, 07eh, 0c3h, 0c0h, 0c0h, 0c0h, 0c3h, 07eh, 018h, 018h, 000h, 000h, 000h, 000h
634 000177DD 7E181800000000      <1>
635 000177E4 00386C6460F0606060- <1>     db  000h, 038h, 06ch, 064h, 060h, 0f0h, 060h, 060h, 060h, 060h, 0e6h, 0fch, 000h, 000h, 000h, 000h
635 000177ED 60E6FC00000000      <1>
636 000177F4 0000C3663C18FF18FF- <1>     db  000h, 000h, 0c3h, 066h, 03ch, 018h, 0ffh, 018h, 0ffh, 018h, 018h, 018h, 000h, 000h, 000h, 000h
636 000177FD 18181800000000      <1>
637 00017804 00FC66667C62666F66- <1>     db  000h, 0fch, 066h, 066h, 07ch, 062h, 066h, 06fh, 066h, 066h, 066h, 0f3h, 000h, 000h, 000h, 000h
637 0001780D 6666F300000000      <1>
638 00017814 000E1B1818187E1818- <1>     db  000h, 00eh, 01bh, 018h, 018h, 018h, 07eh, 018h, 018h, 018h, 018h, 018h, 0d8h, 070h, 000h, 000h
638 0001781D 181818D8700000      <1>
639 00017824 0018306000780C7CCC- <1>     db  000h, 018h, 030h, 060h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
639 0001782D CCCC7600000000      <1>
640 00017834 000C18300038181818- <1>     db  000h, 00ch, 018h, 030h, 000h, 038h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h
640 0001783D 18183C00000000      <1>
641 00017844 00183060007CC6C6C6- <1>     db  000h, 018h, 030h, 060h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
641 0001784D C6C67C00000000      <1>
642 00017854 0018306000CCCCCCCC- <1>     db  000h, 018h, 030h, 060h, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
642 0001785D CCCC7600000000      <1>
643 00017864 000076DC00DC666666- <1>     db  000h, 000h, 076h, 0dch, 000h, 0dch, 066h, 066h, 066h, 066h, 066h, 066h, 000h, 000h, 000h, 000h
643 0001786D 66666600000000      <1>
644 00017874 76DC00C6E6F6FEDECE- <1>     db  076h, 0dch, 000h, 0c6h, 0e6h, 0f6h, 0feh, 0deh, 0ceh, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h, 000h
644 0001787D C6C6C600000000      <1>
645 00017884 003C6C6C3E007E0000- <1>     db  000h, 03ch, 06ch, 06ch, 03eh, 000h, 07eh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
645 0001788D 00000000000000      <1>
646 00017894 00386C6C38007C0000- <1>     db  000h, 038h, 06ch, 06ch, 038h, 000h, 07ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
646 0001789D 00000000000000      <1>
647 000178A4 0000303000303060C0- <1>     db  000h, 000h, 030h, 030h, 000h, 030h, 030h, 060h, 0c0h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
647 000178AD C6C67C00000000      <1>
648 000178B4 000000000000FEC0C0- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 0feh, 0c0h, 0c0h, 0c0h, 0c0h, 000h, 000h, 000h, 000h, 000h
648 000178BD C0C000000000000     <1>
649 000178C4 000000000000FE0606- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 0feh, 006h, 006h, 006h, 006h, 000h, 000h, 000h, 000h, 000h
649 000178CD 06060000000000      <1>
650 000178D4 00C0C0C2C6CC183060- <1>     db  000h, 0c0h, 0c0h, 0c2h, 0c6h, 0cch, 018h, 030h, 060h, 0ceh, 09bh, 006h, 00ch, 01fh, 000h, 000h
650 000178DD CE9B060C1F0000      <1>
651 000178E4 00C0C0C2C6CC183066- <1>     db  000h, 0c0h, 0c0h, 0c2h, 0c6h, 0cch, 018h, 030h, 066h, 0ceh, 096h, 03eh, 006h, 006h, 000h, 000h
651 000178ED CE963E06060000      <1>
652 000178F4 00001818001818183C- <1>     db  000h, 000h, 018h, 018h, 000h, 018h, 018h, 018h, 03ch, 03ch, 03ch, 018h, 000h, 000h, 000h, 000h
652 000178FD 3C3C1800000000      <1>
653 00017904 0000000000366CD86C- <1>     db  000h, 000h, 000h, 000h, 000h, 036h, 06ch, 0d8h, 06ch, 036h, 000h, 000h, 000h, 000h, 000h, 000h
653 0001790D 36000000000000      <1>
654 00017914 0000000000D86C366C- <1>     db  000h, 000h, 000h, 000h, 000h, 0d8h, 06ch, 036h, 06ch, 0d8h, 000h, 000h, 000h, 000h, 000h, 000h
654 0001791D D8000000000000      <1>
655 00017924 114411441144114411- <1>     db  011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h
655 0001792D 44114411441144      <1>
656 00017934 55AA55AA55AA55AA55- <1>     db  055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah
656 0001793D AA55AA55AA55AA      <1>
657 00017944 DD77DD77DD77DD77DD- <1>     db  0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 0ddh, 077h
657 0001794D 77DD77DD77DD77      <1>
658 00017954 1818181818181818- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
658 0001795D 18181818181818      <1>
659 00017964 1818181818818F818- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 0f8h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
659 0001796D 18181818181818      <1>
660 00017974 1818181818F818F818- <1>     db  018h, 018h, 018h, 018h, 018h, 0f8h, 018h, 0f8h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
660 0001797D 18181818181818      <1>
661 00017984 36363636363636F636- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 0f6h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
661 0001798D 36363636363636      <1>
662 00017994 000000000000FE36- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0feh, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
662 0001799D 36363636363636      <1>
663 000179A4 0000000000F818F818- <1>     db  000h, 000h, 000h, 000h, 000h, 0f8h, 018h, 0f8h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
663 000179AD 18181818181818      <1>
664 000179B4 3636363636F606F636- <1>     db  036h, 036h, 036h, 036h, 036h, 0f6h, 006h, 0f6h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
664 000179BD 36363636363636      <1>
665 000179C4 363636363636363636- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
665 000179CD 36363636363636      <1>
666 000179D4 0000000000FE06F636- <1>     db  000h, 000h, 000h, 000h, 000h, 0feh, 006h, 0f6h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
666 000179DD 36363636363636      <1>
667 000179E4 3636363636F606FE00- <1>     db  036h, 036h, 036h, 036h, 036h, 0f6h, 006h, 0feh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
667 000179ED 00000000000000      <1>
668 000179F4 36363636363636FE00- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 0feh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
668 000179FD 00000000000000      <1>
669 00017A04 1818181818F818F800- <1>     db  018h, 018h, 018h, 018h, 018h, 0f8h, 018h, 0f8h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
669 00017A0D 00000000000000      <1>
670 00017A14 000000000000000F818- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0f8h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
670 00017A1D 18181818181818      <1>
671 00017A24 1818181818181F00- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 01fh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
671 00017A2D 00000000000000      <1>
672 00017A34 18181818181818FF00- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
672 00017A3D 00000000000000      <1>
673 00017A44 000000000000000FF18- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
673 00017A4D 18181818181818      <1>
674 00017A54 1818181818181F18- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 01fh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
674 00017A5D 18181818181818      <1>
675 00017A64 000000000000000FF00- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
675 00017A6D 00000000000000      <1>
676 00017A74 1818181818181FF18- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
676 00017A7D 18181818181818      <1>
677 00017A84 1818181818F181F18- <1>     db  018h, 018h, 018h, 018h, 018h, 01fh, 018h, 01fh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
677 00017A8D 18181818181818      <1>
678 00017A94 363636363636363736- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 037h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
678 00017A9D 36363636363636      <1>
679 00017AA4 363636363637303F00- <1>     db  036h, 036h, 036h, 036h, 036h, 037h, 030h, 03fh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
679 00017AAD 00000000000000      <1>
680 00017AB4 00000000003F303736- <1>     db  000h, 000h, 000h, 000h, 000h, 03fh, 030h, 037h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
680 00017ABD 36363636363636      <1>
681 00017AC4 3636363636F700FF00- <1>     db  036h, 036h, 036h, 036h, 036h, 0f7h, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
681 00017ACD 00000000000000      <1>
682 00017AD4 0000000000FF00F736- <1>     db  000h, 000h, 000h, 000h, 000h, 0ffh, 000h, 0f7h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
682 00017ADD 36363636363636      <1>
683 00017AE4 363636363637303736- <1>     db  036h, 036h, 036h, 036h, 036h, 037h, 030h, 037h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
683 00017AED 36363636363636      <1>
684 00017AF4 0000000000FF00FF00- <1>     db  000h, 000h, 000h, 000h, 000h, 0ffh, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
684 00017AFD 00000000000000      <1>
685 00017B04 3636363636F700F736- <1>     db  036h, 036h, 036h, 036h, 036h, 0f7h, 000h, 0f7h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
685 00017B0D 36363636363636      <1>
686 00017B14 1818181818FF00FF00- <1>     db  018h, 018h, 018h, 018h, 018h, 0ffh, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
686 00017B1D 00000000000000      <1>
687 00017B24 36363636363636FF00- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
687 00017B2D 00000000000000      <1>
688 00017B34 0000000000FF00FF18- <1>     db  000h, 000h, 000h, 000h, 000h, 0ffh, 000h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
688 00017B3D 18181818181818      <1>
689 00017B44 000000000000000FF36- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
689 00017B4D 36363636363636      <1>
690 00017B54 363636363636363F00- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 03fh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
```

```
00017B5D 00000000000000      <1>
00017B64 18181818181F181F00- <1>     db  018h, 018h, 018h, 018h, 018h, 01fh, 018h, 01fh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
00017B6D 00000000000000      <1>
00017B74 00000000001F181F18- <1>     db  000h, 000h, 000h, 000h, 000h, 01fh, 018h, 01fh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
00017B7D 18181818181818      <1>
00017B84 0000000000000003F36- <1>    db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 03fh, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
00017B8D 36363636363636      <1>
00017B94 36363636363636FF36- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 0ffh, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
00017B9D 36363636363636      <1>
00017BA4 1818181818FF18FF18- <1>     db  018h, 018h, 018h, 018h, 018h, 0ffh, 018h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
00017BAD 18181818181818      <1>
00017BB4 18181818181818F800- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 0f8h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
00017BBD 00000000000000      <1>
00017BC4 000000000000001F18- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 01fh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
00017BCD 18181818181818      <1>
00017BD4 FFFFFFFFFFFFFFFFFF- <1>     db  0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh
00017BDD FFFFFFFFFFFFFF      <1>
00017BE4 000000000000000FFFF- <1>    db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh
00017BED FFFFFFFFFFFFFF      <1>
00017BF4 F0F0F0F0F0F0F0F0F0- <1>     db  0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h
00017BFD F0F0F0F0F0F0F0      <1>
00017C04 0F0F0F0F0F0F0F0F0F- <1>     db  00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh
00017C0D 0F0F0F0F0F0F0F      <1>
00017C14 FFFFFFFFFFFFFF0000- <1>     db  0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
00017C1D 00000000000000      <1>
00017C24 000000000076DCD8D8- <1>     db  000h, 000h, 000h, 000h, 000h, 076h, 0dch, 0d8h, 0d8h, 0d8h, 0dch, 076h, 000h, 000h, 000h, 000h
00017C2D D8DC7600000000      <1>
00017C34 000078CCCCCCD8CCC6- <1>     db  000h, 000h, 078h, 0cch, 0cch, 0cch, 0d8h, 0cch, 0c6h, 0c6h, 0c6h, 0cch, 000h, 000h, 000h, 000h
00017C3D C6C6CC00000000      <1>
00017C44 0000FEC6C6C0C0C0C0- <1>     db  000h, 000h, 0feh, 0c6h, 0c6h, 0c0h, 0c0h, 0c0h, 0c0h, 0c0h, 0c0h, 0c0h, 000h, 000h, 000h, 000h
00017C4D C0C0C000000000      <1>
00017C54 00000000FE6C6C6C6C- <1>     db  000h, 000h, 000h, 000h, 0feh, 06ch, 06ch, 06ch, 06ch, 06ch, 06ch, 06ch, 000h, 000h, 000h, 000h
00017C5D 6C6C6C00000000      <1>
00017C64 000000FEC660301830- <1>     db  000h, 000h, 000h, 0feh, 0c6h, 060h, 030h, 018h, 030h, 060h, 0c6h, 0feh, 000h, 000h, 000h, 000h
00017C6D 60C6FE00000000      <1>
00017C74 00000000007ED8D8D8- <1>     db  000h, 000h, 000h, 000h, 000h, 07eh, 0d8h, 0d8h, 0d8h, 0d8h, 0d8h, 070h, 000h, 000h, 000h, 000h
00017C7D D8D87000000000      <1>
00017C84 000000006666666666- <1>     db  000h, 000h, 000h, 000h, 066h, 066h, 066h, 066h, 066h, 07ch, 060h, 060h, 0c0h, 000h, 000h, 000h
00017C8D 7C6060C0000000      <1>
00017C94 0000000076DC181818- <1>     db  000h, 000h, 000h, 000h, 076h, 0dch, 018h, 018h, 018h, 018h, 018h, 018h, 000h, 000h, 000h, 000h
00017C9D 18181800000000      <1>
00017CA4 0000007E183C666666- <1>     db  000h, 000h, 000h, 07eh, 018h, 03ch, 066h, 066h, 066h, 03ch, 018h, 07eh, 000h, 000h, 000h, 000h
00017CAD 3C187E00000000      <1>
00017CB4 000000386CC6C6FEC6- <1>     db  000h, 000h, 000h, 038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 06ch, 038h, 000h, 000h, 000h, 000h
00017CBD C66C3800000000      <1>
00017CC4 0000386CC6C6C66C6C- <1>     db  000h, 000h, 038h, 06ch, 0c6h, 0c6h, 0c6h, 06ch, 06ch, 06ch, 06ch, 0eeh, 000h, 000h, 000h, 000h
00017CCD 6C6CEE00000000      <1>
00017CD4 00001E30180C3E6666- <1>     db  000h, 000h, 01eh, 030h, 018h, 00ch, 03eh, 066h, 066h, 066h, 066h, 03ch, 000h, 000h, 000h, 000h
00017CDD 66663C00000000      <1>
00017CE4 00000000007EDBDBDB- <1>     db  000h, 000h, 000h, 000h, 000h, 07eh, 0dbh, 0dbh, 0dbh, 07eh, 000h, 000h, 000h, 000h, 000h, 000h
00017CED 7E000000000000      <1>
00017CF4 00000003067EDBDBF3- <1>     db  000h, 000h, 000h, 003h, 006h, 07eh, 0dbh, 0dbh, 0f3h, 07eh, 060h, 0c0h, 000h, 000h, 000h, 000h
00017CFD 7E60C000000000      <1>
00017D04 00001C3060607C6060- <1>     db  000h, 000h, 01ch, 030h, 060h, 060h, 07ch, 060h, 060h, 060h, 030h, 01ch, 000h, 000h, 000h, 000h
00017D0D 60301C00000000      <1>
00017D14 0000007CC6C6C6C6C6- <1>     db  000h, 000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h, 000h
00017D1D C6C6C600000000      <1>
00017D24 00000000FE0000FE00- <1>     db  000h, 000h, 000h, 000h, 0feh, 000h, 000h, 0feh, 000h, 000h, 0feh, 000h, 000h, 000h, 000h, 000h
00017D2D 00FE0000000000      <1>
00017D34 0000000018187E1818- <1>     db  000h, 000h, 000h, 000h, 018h, 018h, 07eh, 018h, 018h, 000h, 000h, 0ffh, 000h, 000h, 000h, 000h
00017D3D 0000FF00000000      <1>
00017D44 00000030180C060C18- <1>     db  000h, 000h, 000h, 030h, 018h, 00ch, 006h, 00ch, 018h, 030h, 000h, 07eh, 000h, 000h, 000h, 000h
00017D4D 30007E00000000      <1>
00017D54 0000000C1830603018- <1>     db  000h, 000h, 000h, 00ch, 018h, 030h, 060h, 030h, 018h, 00ch, 000h, 07eh, 000h, 000h, 000h, 000h
00017D5D 0C007E00000000      <1>
00017D64 00000E1B1B18181818- <1>     db  000h, 000h, 00eh, 01bh, 01bh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
00017D6D 18181818181818      <1>
00017D74 1818181818181818D8- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 0d8h, 0d8h, 0d8h, 070h, 000h, 000h, 000h, 000h
00017D7D D8D87000000000      <1>
00017D84 000000001818007E00- <1>     db  000h, 000h, 000h, 000h, 018h, 018h, 000h, 07eh, 000h, 018h, 018h, 000h, 000h, 000h, 000h, 000h
00017D8D 18180000000000      <1>
00017D94 000000000076DC0076- <1>     db  000h, 000h, 000h, 000h, 000h, 076h, 0dch, 000h, 076h, 0dch, 000h, 000h, 000h, 000h, 000h, 000h
00017D9D DC000000000000      <1>
00017DA4 00386C6C3800000000- <1>     db  000h, 038h, 06ch, 06ch, 038h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
00017DAD 00000000000000      <1>
00017DB4 000000000000001818- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
00017DBD 00000000000000      <1>
00017DC4 000000000000000018- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
00017DCD 00000000000000      <1>
00017DD4 000F0C0C0C0C0CEC6C- <1>     db  000h, 00fh, 00ch, 00ch, 00ch, 00ch, 00ch, 0ech, 06ch, 06ch, 03ch, 01ch, 000h, 000h, 000h, 000h
00017DDD 6C3C1C00000000      <1>
00017DE4 00D86C6C6C6C6C0000- <1>     db  000h, 0d8h, 06ch, 06ch, 06ch, 06ch, 06ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
00017DED 00000000000000      <1>
00017DF4 0070D83060C8F80000- <1>     db  000h, 070h, 0d8h, 030h, 060h, 0c8h, 0f8h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
00017DFD 00000000000000      <1>
00017E04 000000007C7C7C7C7C- <1>     db  000h, 000h, 000h, 000h, 07ch, 07ch, 07ch, 07ch, 07ch, 07ch, 07ch, 000h, 000h, 000h, 000h, 000h
00017E0D 7C7C0000000000      <1>
00017E14 000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
00017E1D 00000000000000      <1>
                             <1>
                             <1> ; 01/01/2021 (TRDOS 386 v2.0.3)
                             <1>
                             <1> %if 0
                             <1>
                             <1> vgafont14alt:
                             <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, 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, 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, 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, 000h, 059h, 000h, 000h, 0c3h, 0c3h, 0c3h, 066h, 03ch, 018h
                        <1>     db  018h, 018h, 018h, 03ch, 000h, 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, 077h, 000h, 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"
00017E24 424F4348532F51454D-                    db "BOCHS/QEMU/VIRTUALBOX"  ; 26/11/2020
00017E2D 552F5649525455414C-
00017E36 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 ..."
00017E39 205645534120564245-                  db " VESA VBE3 Video BIOS ..." ; 26/11/2020
00017E42 3320566964656F2042-
00017E4B 494F53202E2E2E
00017E52 0D0A0D0A00                           db 0Dh, 0Ah, 0Dh, 0Ah, 0

00017E57 90                             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
                                        ;
00017E58 B207                           year:       dw 1970
00017E5A 0100                           month:      dw 1
00017E5C 0100                           day:  dw 1
00017E5E 0000                           hour:       dw 0
00017E60 0000                           minute: dw 0
00017E62 0000                           second: dw 0

                                        DMonth:
00017E64 0000                                 dw 0
00017E66 1F00                                 dw 31
00017E68 3B00                                 dw 59
00017E6A 5A00                                 dw 90
00017E6C 7800                                 dw 120
00017E6E 9700                                 dw 151
00017E70 B500                                 dw 181
00017E72 D400                                 dw 212
00017E74 F300                                 dw 243
00017E76 1101                                 dw 273
00017E78 3001                                 dw 304
00017E7A 4E01                                 dw 334

                                        ; 15/11/2020
00017E7C 00                             db    0
                                        kernel_version_msg: ; 20/11/2020
00017E7D 5452444F5320283338-            db    "TRDOS (386) Kernel v2.0.3 by Erdogan Tan"
00017E86 3629204B65726E656C-
00017E8F 2076322E302E332062-
00017E98 79204572646F67616E-
00017EA1 2054616E
00017EA5 00                             db    0

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

                                        bss_start:

                                        ABSOLUTE bss_start

00017EA6 <res 00000002>                 alignb 8 ; 25/12/2016

                                              ; 15/04/2016
                                              ; TRDOS 386 (TRDOS v2.0)
                                              ;     80 interrupts
                                              ; 11/03/2015
                                              ; Interrupt Descriptor Table (20/08/2014)
                                        idt:
                                              ;resb  64*8 ; INT 0 to INT 3Fh
                                              ; 15/04/2016
00017EA8 <res 00000280>                       resb   80*8 ; INT 0 to INT 4Fh

                                        idt_end:

                                        ;alignb 4

                                        task_state_segment:
                                              ; 24/03/2015
00018128 <res 00000002>                 tss.link:   resw 1
0001812A <res 00000002>                           resw 1
                                        ; tss offset 4
0001812C <res 00000004>                 tss.esp0:   resd 1
00018130 <res 00000002>                 tss.ss0:    resw 1
00018132 <res 00000002>                           resw 1
```

```
3508 00018134 <res 00000004>          tss.esp1:   resd 1
3509 00018138 <res 00000002>          tss.ss1:    resw 1
3510 0001813A <res 00000002>                      resw 1
3511 0001813C <res 00000004>          tss.esp2:   resd 1
3512 00018140 <res 00000002>          tss.ss2:    resw 1
3513 00018142 <res 00000002>                      resw 1
3514                                  ; tss offset 28
3515 00018144 <res 00000004>          tss.CR3:    resd 1
3516 00018148 <res 00000004>          tss.eip:    resd 1
3517 0001814C <res 00000004>          tss.eflags: resd 1
3518                                  ; tss offset 40
3519 00018150 <res 00000004>          tss.eax:    resd 1
3520 00018154 <res 00000004>          tss.ecx:    resd 1
3521 00018158 <res 00000004>          tss.edx:    resd 1
3522 0001815C <res 00000004>          tss.ebx:    resd 1
3523 00018160 <res 00000004>          tss.esp:    resd 1
3524 00018164 <res 00000004>          tss.ebp:    resd 1
3525 00018168 <res 00000004>          tss.esi:    resd 1
3526 0001816C <res 00000004>          tss.edi:    resd 1
3527                                  ; tss offset 72
3528 00018170 <res 00000002>          tss.ES:     resw 1
3529 00018172 <res 00000002>                      resw 1
3530 00018174 <res 00000002>          tss.CS:         resw 1
3531 00018176 <res 00000002>                      resw 1
3532 00018178 <res 00000002>          tss.SS:         resw 1
3533 0001817A <res 00000002>                      resw 1
3534 0001817C <res 00000002>          tss.DS:         resw 1
3535 0001817E <res 00000002>                      resw 1
3536 00018180 <res 00000002>          tss.FS:         resw 1
3537 00018182 <res 00000002>                      resw 1
3538 00018184 <res 00000002>          tss.GS:         resw 1
3539 00018186 <res 00000002>                      resw 1
3540 00018188 <res 00000002>          tss.LDTR:   resw 1
3541 0001818A <res 00000002>                      resw 1
3542                                  ; tss offset 100
3543 0001818C <res 00000002>                      resw 1
3544 0001818E <res 00000002>          tss.IOPB:   resw 1
3545                                  ; tss offset 104
3546                                  tss_end:
3547
3548 00018190 <res 00000004>          k_page_dir:  resd 1 ; Kernel's (System) Page Directory address
3549                                                      ; (Physical address = Virtual address)
3550 00018194 <res 00000004>          memory_size: resd 1 ; memory size in pages
3551 00018198 <res 00000004>          free_pages:  resd 1 ; number of free pages
3552 0001819C <res 00000004>          next_page:   resd 1 ; offset value in M.A.T. for
3553                                                      ; first free page search
3554 000181A0 <res 00000004>          last_page:   resd 1 ; offset value in M.A.T. which
3555                                                      ; next free page search will be
3556                                                      ; stopped after it. (end of M.A.T.)
3557 000181A4 <res 00000004>          first_page:  resd 1 ; offset value in M.A.T. which
3558                                                      ; first free page search
3559                                                      ; will be started on it. (for user)
3560 000181A8 <res 00000004>          mat_size:    resd 1 ; Memory Allocation Table size in pages
3561
3562                                  ; 20/11/2020
3563                                  ;vbe2bios:    resw 1 ; VBE2 video bios ID (bochs/qemu)
3564                                  ;               ; (0B0C4h or 0B0C5h for bochs/plex86 vgabios)
3565
3566                                  ; 02/09/2014 (Retro UNIX 386 v1)
3567                                  ; 04/12/2013 (Retro UNIX 8086 v1)
3568 000181AC <res 00000002>          CRT_START:   resw 1         ; starting address in regen buffer
3569                                                     ; NOTE: active page only
3570 000181AE <res 00000010>          CURSOR_POSN: resw 8 ; cursor positions for video pages
3571                                  ACTIVE_PAGE:
3572 000181BE <res 00000001>          ptty:            resb 1 ; current tty
3573                                  ; 01/07/2015 - 29/01/2016
3574 000181BF <res 00000001>          ccolor:          resb 1 ; current color attribute
3575                                  ; 26/10/2015
3576                                  ; 07/09/2014
3577 000181C0 <res 00000014>          ttychr:      resw ntty+2 ; Character buffer (multiscreen)
3578
3579                                  ; 18/05/2015 (03/06/2013 - Retro UNIX 8086 v1 feature only!)
3580 000181D4 <res 00000004>          p_time:      resd 1     ; present time (for systime & sysmdate)
3581
3582                                  ; 18/05/2015 (16/08/2013 - Retro UNIX 8086 v1 feature only !)
3583                                  ; (open mode locks for pseudo TTYs)
3584                                  ; [ major tty locks (return error in any conflicts) ]
3585 000181D8 <res 00000014>          ttyl:        resw ntty+2 ; opening locks for TTYs.
3586
3587                                  ; 15/04/2015 (Retro UNIX 386 v1)
3588                                  ; 22/09/2013 (Retro UNIX 8086 v1)
3589 000181EC <res 0000000A>          wlist:       resb ntty+2 ; wait channel list (0 to 9 for TTYs)
3590                                  ; 15/04/2015 (Retro UNIX 386 v1)
3591                                  ;; 12/07/2014 -> sp_init set comm. parameters as 0E3h
3592                                  ;; 0 means serial port is not available
3593                                  ;;comprm: ; 25/06/2014
3594 000181F6 <res 00000001>          com1p:       resb 1  ;;0E3h
3595 000181F7 <res 00000001>          com2p:       resb 1  ;;0E3h
3596
3597                                  ; 17/11/2015
3598                                  ; request for response (from the terminal)
3599 000181F8 <res 00000002>          req_resp:    resw 1
3600                                  ; 07/11/2015
3601 000181FA <res 00000001>          ccomport:    resb 1 ; current COM (serial) port
3602                                                   ; (0= COM1, 1= COM2)
3603                                  ; 09/11/2015
3604 000181FB <res 00000001>          comqr:           resb 1 ; 'query or response' sign (u9.s, 'sndc')
3605                                  ; 07/11/2015
3606 000181FC <res 00000002>          rchar:           resw 1 ; last received char for COM 1 and COM 2
3607 000181FE <res 00000002>          schar:           resw 1 ; last sent char for COM 1 and COM 2
3608
3609                                  ; 22/08/2014 (RTC)
3610                                  ; (Packed BCD)
3611 00018200 <res 00000001>          time_seconds: resb 1
3612 00018201 <res 00000001>          time_minutes: resb 1
```

```
3613 00018202 <res 00000001>          time_hours:   resb 1
3614 00018203 <res 00000001>          date_wday:    resb 1
3615 00018204 <res 00000001>          date_day:     resb 1
3616 00018205 <res 00000001>          date_month:   resb 1
3617 00018206 <res 00000001>          date_year:    resb 1
3618 00018207 <res 00000001>          date_century: resb 1
3619
3620                                  ; 24/01/2016
3621 00018208 <res 00000004>          RTC_LH:            resd 1
3622 0001820C <res 00000001>          RTC_WAIT_FLAG: resb 1
3623 0001820D <res 00000001>          USER_FLAG:     resb 1
3624                                  ; 19/05/2016
3625                                  ;RTC_second:
3626 0001820E <res 00000001>          RTC_2Hz:       resb 1 ;  from 2Hz interrupt to 1Hz timer event function
3627
3628                                  %include 'diskbss.s'      ; UNINITIALIZED DISK (BIOS) DATA
   1                                <1> ; ****************************************************************************
   2                                <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.0 - diskbss.s
   3                                <1> ; ----------------------------------------------------------------------------
   4                                <1> ; Last Update: 24/01/2016
   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> ; diskbss.inc (10/07/2015)
  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 - DISKBSS.INC
  20                                <1> ; Last Modification: 10/07/2015
  21                                <1> ;     (Unnitialized Disk Parameters Data section for 'DISKIO.INC')
  22                                <1>
  23 0001820F <res 00000001>        <1> alignb 2
  24                                <1>
  25                                <1> ;----------------------------------------
  26                                <1> ;     TIMER DATA AREA           :
  27                                <1> ;----------------------------------------
  28                                <1>
  29                                <1> TIMER_LH:    ; 16/02/205
  30 00018210 <res 00000002>        <1> TIMER_LOW:      resw     1               ; LOW WORD OF TIMER COUNT
  31 00018212 <res 00000002>        <1> TIMER_HIGH:     resw     1               ; HIGH WORD OF TIMER COUNT
  32 00018214 <res 00000001>        <1> TIMER_OFL:      resb     1               ; TIMER HAS ROLLED OVER SINCE LAST READ
  33                                <1>
  34                                <1> ;----------------------------------------
  35                                <1> ;     DISKETTE DATA AREAS       :
  36                                <1> ;----------------------------------------
  37                                <1>
  38 00018215 <res 00000001>        <1> SEEK_STATUS: resb   1
  39 00018216 <res 00000001>        <1> MOTOR_STATUS:      resb   1
  40 00018217 <res 00000001>        <1> MOTOR_COUNT: resb   1
  41 00018218 <res 00000001>        <1> DSKETTE_STATUS:    resb   1
  42 00018219 <res 00000007>        <1> NEC_STATUS:  resb   7
  43                                <1>
  44                                <1> ;----------------------------------------
  45                                <1> ;     ADDITIONAL MEDIA DATA           :
  46                                <1> ;----------------------------------------
  47                                <1>
  48 00018220 <res 00000001>        <1> LASTRATE:    resb   1
  49 00018221 <res 00000001>        <1> HF_STATUS:   resb   1
  50 00018222 <res 00000001>        <1> HF_ERROR:    resb   1
  51 00018223 <res 00000001>        <1> HF_INT_FLAG: resb   1
  52 00018224 <res 00000001>        <1> HF_CNTRL:    resb   1
  53 00018225 <res 00000004>        <1> DSK_STATE:   resb   4
  54 00018229 <res 00000002>        <1> DSK_TRK:     resb   2
  55                                <1>
  56                                <1> ;----------------------------------------
  57                                <1> ;     FIXED DISK DATA AREAS           :
  58                                <1> ;----------------------------------------
  59                                <1>
  60 0001822B <res 00000001>        <1> DISK_STATUS1:      resb   1            ; FIXED DISK STATUS
  61 0001822C <res 00000001>        <1> HF_NUM:            resb   1            ; COUNT OF FIXED DISK DRIVES
  62 0001822D <res 00000001>        <1> CONTROL_BYTE:      resb   1            ; HEAD CONTROL BYTE
  63                                <1> ;@PORT_OFF   resb   1            ; RESERVED (PORT OFFSET)
  64                                <1> ;port1_off   resb   1            ; Hard disk controller 1 - port offset
  65                                <1> ;port2_off   resb   1            ; Hard idsk controller 2 - port offset
  66                                <1>
  67 0001822E <res 00000002>        <1> alignb 4
  68                                <1>
  69                                <1> ;HF_TBL_VEC: resd   1            ; Primary master disk param. tbl. pointer
  70                                <1> ;HF1_TBL_VEC:      resd   1            ; Primary slave disk param. tbl. pointer
  71                                <1> HF_TBL_VEC: ; 22/12/2014
  72 00018230 <res 00000004>        <1> HDPM_TBL_VEC:      resd   1            ; Primary master disk param. tbl. pointer
  73 00018234 <res 00000004>        <1> HDPS_TBL_VEC:      resd   1            ; Primary slave disk param. tbl. pointer
  74 00018238 <res 00000004>        <1> HDSM_TBL_VEC:      resd   1            ; Secondary master disk param. tbl. pointer
  75 0001823C <res 00000004>        <1> HDSS_TBL_VEC:      resd   1            ; Secondary slave disk param. tbl. pointer
  76                                <1>
  77                                <1> ; 03/01/2015
  78 00018240 <res 00000001>        <1> LBAMode:           resb   1
  79                                <1>
  80                                <1> ; ****************************************************************************
3629
3630                                    ;;; Real Mode Data (10/07/2015 - BSS)
3631
3632                                    ;alignb 2
3633
3634                                    ; 10/01/2016
3635                                    %include 'trdoskx.s'      ; UNINITIALIZED KERNEL (Logical Drive & FS) DATA
   1                                <1> ; ****************************************************************************
   2                                <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>
00018241 <res 00000003>     <1> alignb 4
                            <1>
                            <1> ; MAINPROG.ASM
00018244 <res 00000004>     <1> MainProgCfg_FileSize:   resd 1 ; 14/04/2016
00018248 <res 00000004>     <1> MainProgCfg_LineOffset: resd 1 ; 14/04/2016
                            <1>
0001824C <res 00000004>     <1> Current_VolSerial: resd 1
                            <1>
00018250 <res 00000004>     <1> Current_Dir_FCluster: resd 1
                            <1>
00018254 <res 00000001>     <1> Current_Dir_Level: resb 1
00018255 <res 00000001>     <1> Current_FATType: resb 1
00018256 <res 00000001>     <1> Current_Drv: resb 1
00018257 <res 00000001>     <1> Current_Dir_Drv:   resb 1 ; '?'
00018258 <res 00000001>     <1>                    resb 1 ; ':'
00018259 <res 00000001>     <1> Current_Dir_Root:  resb 1 ; '/'
0001825A <res 0000005A>     <1> Current_Directory: resb 90
000182B4 <res 00000001>     <1> End_Of_Current_Dir_Str: resb 1
000182B5 <res 00000001>     <1> Current_Dir_StrLen: resb 1
                            <1>
000182B6 <res 00000001>     <1> CursorColumn:      resb 1
000182B7 <res 00000001>     <1> CmdArgStart:    resb 1
                            <1>
                            <1> ; 03/02/2016
000182B8 <res 0000004E>     <1> Remark:            resb 78
                            <1>
00018306 <res 00000050>     <1> CommandBuffer:     resb 80
                            <1>
00018356 <res 00000100>     <1> TextBuffer:  resb 256
                            <1>
                            <1> MasterBootBuff:
00018456 <res 000001BE>     <1> MasterBootCode: resb 1BEh
00018614 <res 00000040>     <1> PartitionTable: resb 64
00018654 <res 00000002>     <1> MBIDCode: resw 1
                            <1>
                            <1> PTable_Buffer:
00018656 <res 00000040>     <1> PTable_hd0: resb 64
00018696 <res 00000040>     <1> PTable_hd1: resb 64
000186D6 <res 00000040>     <1> PTable_hd2: resb 64
00018716 <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
00018756 <res 00000001>     <1> scount: resb 1 ; 16/05/2016 (diskio.s, 'int33h:')
00018757 <res 00000001>     <1>       resb 1
00018758 <res 00000001>     <1>       resb 1
00018759 <res 00000001>     <1>       resb 1
                            <1>
0001875A <res 00000001>     <1> HD_LBA_yes: resb 1
0001875B <res 00000001>     <1> PP_Counter: resb 1
0001875C <res 00000001>     <1> EP_Counter: resb 1
                            <1> ; 13/08/2020
0001875D <res 00000001>     <1> LD_Counter: resb 1
                            <1>
                            <1> ; 30/08/2020
0001875E <res 00000004>     <1> MBR_EP_StartSector: resd 1
                            <1>             ; Extd partition start sector as in MBR
00018762 <res 00000004>     <1> EP_StartSector: resd 1 ; next extd partition start sector
                            <1>       ; 15/07/2020
                            <1>                ;resd 1
                            <1>                ;resd 1
                            <1>
                            <1> ; 20/07/2020
00018766 <res 00000200>     <1> DOSBootSectorBuff: resb 512
                            <1> ; 15/07/2020
                            <1> ;DOSBootSectorBuff: resb 446 ; 1BEh
                            <1> ;MiniPartitionTable: resb 64 ;  40h
                            <1> ;MiniPartitionMagic: resw  1 ;  02h
                            <1>
                            <1> FAT_BuffDescriptor:
00018966 <res 00000004>     <1> FAT_CurrentCluster: resd 1
0001896A <res 00000001>     <1> FAT_BuffValidData: resb 1
0001896B <res 00000001>     <1> FAT_BuffDrvName: resb 1
0001896C <res 00000002>     <1> FAT_BuffOffset: resw 1
0001896E <res 00000004>     <1> FAT_BuffSector: resd 1
                            <1>
00018972 <res 00000004>     <1> FAT_ClusterCounter: resd 1
00018976 <res 00000004>     <1> LastCluster: resd 1
                            <1>
                            <1> ; 16/05/2016
                            <1> ;; 18/03/2016 (TRDOS v2.0)
                            <1> ;ClusterBuffer_Valid: resb 1
                            <1>
                            <1> Dir_BuffDescriptor:
0001897A <res 00000001>     <1> DirBuff_DRV: resb 1
```

```
00018981 <res 00000004>      <1> DirBuff_Cluster: resd 1
00018985 <res 00000002>      <1> DirBuffer_Size: resw 1
                             <1> ;DirBuff_EntryCounter: resw 1
                             <1>
                             <1> ; 01/02/2016
                             <1> ; these are on (real mode) segment 8000h and later
                             <1> ; FAT_Buffer:      resb 1536 ; 3 sectors
                             <1> ; Dir_Buffer:      resb 512*32
                             <1> ; Logical_DOSDisks:  resb 6656 ; 26 * 256 bytes
                             <1>
                             <1> ; 18/01/2016
                             <1>
00018987 <res 00000004>      <1> FreeClusterCount: resd 1
                             <1>
0001898B <res 00000004>      <1> VolSize_Unit1:   resd 1
0001898F <res 00000004>      <1> VolSize_Unit2:   resd 1
                             <1>
00018993 <res 00000004>      <1> Vol_Tot_Sec_Str_Start:       resd 1
00018997 <res 0000000A>      <1> Vol_Tot_Sec_Str:       resb 10
000189A1 <res 00000001>      <1> Vol_Tot_Sec_Str_End:         resb 1
000189A2 <res 00000001>      <1> resb 1
000189A3 <res 00000004>      <1> Vol_Free_Sectors_Str_Start: resd 1
000189A7 <res 0000000A>      <1> Vol_Free_Sectors_Str:        resb 10
000189B1 <res 00000001>      <1> Vol_Free_Sectors_Str_End:   resb 1
                             <1>
                             <1> ; 10/02/2016
000189B2 <res 00000001>      <1> RUN_CDRV: resb 1 ; CMD_INTR.ASM  ; 09/11/2011
                             <1>
                             <1> ; 24/01/2016
000189B3 <res 00000080>      <1> PATH_Array:     resb 128 ; DIR.ASM ; 09/10/2011
                             <1> ; 06/02/2016
00018A33 <res 00000004>      <1> CCD_DriveDT: resd 1 ; DIR.ASM ; (word)
00018A37 <res 00000001>      <1> CCD_Level:  resb 1 ; DIR.ASM
00018A38 <res 00000001>      <1> Last_Dir_Level:    resb 1 ; DIR.ASM
                             <1> ;
00018A39 <res 00000002>      <1> CDLF_AttributesMask: resw 1 ; DIR.ASM
00018A3B <res 00000004>      <1> CDLF_FNAddress:    resd 1 ; DIR.ASM (word)
00018A3F <res 00000002>      <1> CDLF_DEType: resw 1 ; DIR.ASM
                             <1> ;
00018A41 <res 00000001>      <1> CD_COMMAND:  resb 1 ; DIR.ASM
                             <1>
00018A42 <res 00000002>      <1> alignb 4
                             <1>
                             <1> ; 29/01/2016
00018A44 <res 00000001>      <1> Program_Exit:      resb 1 ; CMD_INTR.ASM  ; 09/11/2011
                             <1>
                             <1> ;alignb 4
                             <1> ; 23/02/2016
00018A45 <res 00000001>      <1> disk_rw_op:  resb 1 ;  0 = disk read, 1 = disk write
                             <1> ;disk_rw_spt:      resb 1 ; sectors per track (<= 63) /// (<256)
                             <1> ; 31/01/2016
00018A46 <res 00000001>      <1> retry_count:       resb 1 ; DISK_IO.ASM ; 20/07/2011 (CHS_RetryCount)
00018A47 <res 00000001>      <1> disk_rw_err:       resb 1 ; DISK_IO.ASM ; (Disk_IO_err_code)
00018A48 <res 00000004>      <1> sector_count:      resd 1 ; DISK_IO.ASM ; (Disk_RW_SectorCount)
                             <1>
                             <1> ; 06/02/2016 (long name)
00018A4C <res 00000002>      <1> FDE_AttrMask:         resw 1 ; DIR.ASM
00018A4E <res 00000002>      <1> AmbiguousFileName: resw 1 ; DIR.ASM
00018A50 <res 00000001>      <1> PreviousAttr:         resb 1 ; DIR.ASM
                             <1> ;
00018A51 <res 00000001>      <1> LongNameFound:   resb 1     ; DIR.ASM
00018A52 <res 00000001>      <1> LFN_EntryLength: resb 1   ; DIR.ASM
00018A53 <res 00000001>      <1> LFN_CheckSum:    resb 1   ; DIR.ASM
00018A54 <res 00000084>      <1> LongFileName:    resb 132 ; DIR.ASM
                             <1>
                             <1> ;PATH_Array_Ptr: resw 1 ; DIR.ASM
00018AD8 <res 00000001>      <1> PATH_CDLevel:       resb 1 ; DIR.ASM
00018AD9 <res 00000001>      <1> PATH_Level:   resb 1 ; DIR.ASM
                             <1>
                             <1> ; 07/02/2016
00018ADA <res 0000000D>      <1> Dir_File_Name:     resb 13 ; DIR.ASM ; 09/10/2011
                             <1>
                             <1> ; 10/02/2016
00018AE7 <res 0000000D>      <1> Dir_Entry_Name:    resb 13 ; DIR.ASM
                             <1>
                             <1> alignb 2
                             <1>
00018AF4 <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
00018AF6 <res 00000001>      <1> FindFile_Drv:               resb 1
00018AF7 <res 00000041>      <1> FindFile_Directory:         resb 65
00018B38 <res 0000000D>      <1> FindFile_Name:              resb 13
                             <1> FindFile_LongNameEntryLength:
00018B45 <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
00018B46 <res 00000002>      <1> FindFile_AttributesMask:  resw 1
00018B48 <res 00000020>      <1> FindFile_DirEntry:   resb 32
00018B68 <res 00000004>      <1> FindFile_DirFirstCluster: resd 1
00018B6C <res 00000004>      <1> FindFile_DirCluster:        resd 1
00018B70 <res 00000002>      <1> FindFile_DirEntryNumber:  resw 1
00018B72 <res 00000002>      <1> FindFile_MatchCounter:      resw 1
00018B74 <res 00000002>      <1> FindFile_Reserved:   resw 1 ; 06/03/2016
                             <1>
00018B76 <res 00000004>      <1> First_Path_Pos: resd 1    ; DIR.ASM ; 09/10/2011
00018B7A <res 00000004>      <1> Last_Slash_Pos: resd 1    ; DIR.ASM
                             <1>
                             <1> ; 10/02/2016
```

```
00018B7E <res 00000002>      <1> File_Count:     resw 1    ; DIR.ASM ; 09/10/2011
00018B80 <res 00000002>      <1> Dir_Count:      resw 1
00018B82 <res 00000004>      <1> Total_FSize:    resd 1
00018B86 <res 00000004>      <1> TFS_Dec_Begin:  resd 1
00018B8A <res 0000000A>      <1>                 resb 10
00018B94 <res 00000001>      <1> TFS_Dec_End:    resb 1
                             <1>
00018B95 <res 00000001>      <1> PrintDir_RowCounter: resb 1
                             <1>
00018B96 <res 00000002>      <1> alignb 4
                             <1> ; 15/02/2015 ('show' command variables)
00018B98 <res 00000004>      <1> Show_FDT:    resd 1
00018B9C <res 00000004>      <1> Show_LDDDT:  resd 1
00018BA0 <res 00000004>      <1> Show_Cluster:      resd 1
00018BA4 <res 00000004>      <1> Show_FileSize:     resd 1
00018BA8 <res 00000004>      <1> Show_FilePointer: resd 1
00018BAC <res 00000002>      <1> Show_ClusterPointer: resw 1
00018BAE <res 00000002>      <1> Show_ClusterSize: resw 1
00018BB0 <res 00000001>      <1> Show_RowCount:     resb 1
                             <1>
00018BB1 <res 00000003>      <1> alignb 4
                             <1> ; 21/02/2016
00018BB4 <res 00000004>      <1> DelFile_FNPointer: resd 1 ; ; CMD_INTR.ASM (word) ; 09/11/2011
                             <1> ; 27/02/2016
                             <1> ; DIR.ASM (09/10/2011)
00018BB8 <res 00000004>      <1> DelFile_FCluster:  resd 1
00018BBC <res 00000002>      <1> DelFile_EntryCounter:    resw 1
00018BBE <res 00000001>      <1> DelFile_LNEL:            resb 1
00018BBF <res 00000001>      <1> resb 1
                             <1>
                             <1> ; DIR.ASM
00018BC0 <res 00000004>      <1> mkdir_DirName_Offset:    resd 1
00018BC4 <res 00000004>      <1> mkdir_FFCluster:   resd 1
00018BC8 <res 00000004>      <1> mkdir_LastDirCluster:    resd 1
00018BCC <res 00000004>      <1> mkdir_FreeSectors: resd 1
00018BD0 <res 00000002>      <1> mkdir_attrib:            resw 1
00018BD2 <res 00000001>      <1> mkdir_SecPerClust: resb 1
00018BD3 <res 00000001>      <1> mkdir_add_new_cluster:   resb 1
00018BD4 <res 0000000D>      <1> mkdir_Name:        resb 13
00018BE1 <res 00000002>      <1> resw 1 ; 01/03/2016
                             <1> ; 27/02/2016
00018BE3 <res 00000001>      <1> RmDir_MultiClusters:     resb 1
00018BE4 <res 00000004>      <1> RmDir_DirEntryOffset:    resd 1 ; 01/03/2016 (word -> dword)
00018BE8 <res 00000004>      <1> RmDir_ParentDirCluster: resd 1
00018BEC <res 00000004>      <1> RmDir_DirLastCluster:   resd 1
00018BF0 <res 00000004>      <1> RmDir_PreviousCluster:  resd 1
                             <1> ; 22/02/2016
00018BF4 <res 00000001>      <1> UPDLMDT_CDirLevel: resb 1
00018BF5 <res 00000004>      <1> UPDLMDT_CDirFCluster:    resd 1
                             <1>
00018BF9 <res 00000003>      <1> alignb 4
                             <1> ; DRV_FAT.ASM ; 21/08/2011
00018BFC <res 00000004>      <1> gffc_next_free_cluster:  resd 1
00018C00 <res 00000004>      <1> gffc_first_free_cluster: resd 1
00018C04 <res 00000004>      <1> gffc_last_free_cluster:  resd 1
                             <1>
                             <1> ;29/04/2016
                             <1> Cluster_Index: ; resd 1
                             <1> ; 22/02/2016
00018C08 <res 00000004>      <1> ClusterValue:      resd 1
                             <1> ; 04/03/2016
00018C0C <res 00000001>      <1> Attributes: resb 1
                             <1> ;;CFS_error:  resb 1 ;; 01/03/2016
00018C0D <res 00000001>      <1> resb 1
00018C0E <res 00000001>      <1> CFS_OPType: resb 1
00018C0F <res 00000001>      <1> CFS_Drv:    resb 1
00018C10 <res 00000004>      <1> CFS_CC:          resd 1
00018C14 <res 00000004>      <1> CFS_FAT32FSINFOSEC: resd 1
00018C18 <res 00000004>      <1> CFS_FAT32FC: resd 1
                             <1>
                             <1> ; 27/02/2016
                             <1> ;alignb 4
00018C1C <res 00000004>      <1> glc_prevcluster: resd 1 ; DRV_FAT.ASM (21/08/2011)
                             <1> ; 22/10/2016
00018C20 <res 00000004>      <1> glc_index:   resd 1 ;  Last Cluster Index (22/10/2016)
                             <1>
                             <1> ; DIR.ASM
00018C24 <res 00000002>      <1> DLN_EntryNumber: resw 1
00018C26 <res 00000001>      <1> DLN_40h:      resb 1
                             <1> ; 28/02/2016
00018C27 <res 00000001>      <1> TCC_FATErr:   resb 1 ; DRV_FAT.ASM
                             <1>
                             <1> alignb 4
                             <1> ; DIR.ASM (09/10/2011)
00018C28 <res 00000002>      <1> LCDE_EntryIndex: resw 1 ; LCDE_EntryOffset
00018C2A <res 00000002>      <1> LCDE_ClusterSN:  resw 1
00018C2C <res 00000004>      <1> LCDE_Cluster:       resd 1
00018C30 <res 00000004>      <1> LCDE_ByteOffset: resd 1
                             <1>
                             <1> ;alignb4
                             <1> ; 06/03/2016 (word -> dword)
                             <1> ; CMD_INTR.ASM (01/08/2010)
00018C34 <res 00000004>      <1> SourceFilePath:         resd 1
00018C38 <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)
00018C3C <res 00000001>      <1> SourceFile_Drv:                 resb 1
00018C3D <res 00000041>      <1> SourceFile_Directory:           resb 65
00018C7E <res 0000000D>      <1> SourceFile_Name:          resb 13
                             <1> SourceFile_LongNameEntryLength:
00018C8B <res 00000001>      <1> SourceFile_LongNameYes:         resb 1 ; Sign for longname procedures
                             <1> ;Above 80 bytes
```

```
                            <1> ;is TR-DOS Source File FullName Format/Structure
00018C8C <res 00000002>     <1> SourceFile_AttributesMask:      resw 1
00018C8E <res 00000020>     <1> SourceFile_DirEntry:            resb 32
00018CAE <res 00000004>     <1> SourceFile_DirFirstCluster:     resd 1
00018CB2 <res 00000004>     <1> SourceFile_DirCluster:          resd 1
00018CB6 <res 00000002>     <1> SourceFile_DirEntryNumber:      resw 1
00018CB8 <res 00000002>     <1> SourceFile_MatchCounter:  resw 1
                            <1> ; 16/03/2016
00018CBA <res 00000001>     <1> SourceFile_SecPerClust:         resb 1
00018CBB <res 00000001>     <1> SourceFile_Reserved:            resb 1
                            <1> ; Above is 128 bytes
                            <1>
                            <1> ;Destination File Structure (same with 'Find File' Structure)
00018CBC <res 00000001>     <1> DestinationFile_Drv:            resb 1
00018CBD <res 00000041>     <1> DestinationFile_Directory:      resb 65
00018CFE <res 0000000D>     <1> DestinationFile_Name:           resb 13
                            <1> DestinationFile_LongNameEntryLength:
00018D0B <res 00000001>     <1> DestinationFile_LongNameYes:    resb 1 ; Sign for longname procedures
                            <1> ;Above 80 bytes
                            <1> ;is TR-DOS Destination File FullName Format/Structure
00018D0C <res 00000002>     <1> DestinationFile_AttributesMask: resw 1
00018D0E <res 00000020>     <1> DestinationFile_DirEntry: resb 32
00018D2E <res 00000004>     <1> DestinationFile_DirFirstCluster: resd 1
00018D32 <res 00000004>     <1> DestinationFile_DirCluster:     resd 1
00018D36 <res 00000002>     <1> DestinationFile_DirEntryNumber: resw 1
00018D38 <res 00000002>     <1> DestinationFile_MatchCounter:   resw 1
                            <1> ; 16/03/2016
00018D3A <res 00000001>     <1> DestinationFile_SecPerClust:    resb 1
00018D3B <res 00000001>     <1> DestinationFile_Reserved: resb 1
                            <1> ; Above is 128 bytes
                            <1>
                            <1> ; 24/04/2016
00018D3C <res 00000002>     <1> resw 1
                            <1>
                            <1> ; 10/03/2016
                            <1> ; FILE.ASM
00018D3E <res 00000001>     <1> move_cmd_phase:       resb 1
00018D3F <res 00000001>     <1> msftdf_sf_df_drv:  resb 1
00018D40 <res 00000004>     <1> msftdf_drv_offset: resd 1
                            <1>
                            <1> ; 11/03/2016
                            <1> ; DRV_FAT.ASM (21/08/2011)
00018D44 <res 00000004>     <1> FAT_anc_LCluster:  resd 1
00018D48 <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'
00018D4C <res 00000004>     <1> mem_ipg_count:     resd 1 ; page count (for contiguous allocation)
00018D50 <res 00000004>     <1> mem_pg_count:      resd 1 ; page count (for count down)
00018D54 <res 00000004>     <1> mem_aperture:      resd 1 ; contiguous free pages (current)
00018D58 <res 00000004>     <1> mem_max_aperture: resd 1 ; maximum value of contiguous free pages
00018D5C <res 00000004>     <1> mem_pg_pos:  resd 1 ; mem. position (page #) of current aperture
00018D60 <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')
00018D64 <res 00000001>     <1> copy_cmd_phase:       resb 1
00018D65 <res 00000001>     <1> csftdf_rw_err:           resb 1
00018D66 <res 00000001>     <1> DestinationFileFound: resb 1
00018D67 <res 00000001>     <1> csftdf_cdrv:             resb 1
00018D68 <res 00000004>     <1> csftdf_filesize:      resd 1
                            <1> ; TRDOS386 (TRDOS v2.0)
00018D6C <res 00000004>     <1> csftdf_sf_mem_addr:   resd 1
00018D70 <res 00000004>     <1> csftdf_sf_mem_bsize:  resd 1
                            <1> ;
                            <1>
00018D74 <res 00000004>     <1> csftdf_sf_cluster:    resd 1 ; 16/03/2016
00018D78 <res 00000004>     <1> csftdf_df_cluster:    resd 1
                            <1> ; 16/03/2016
00018D7C <res 00000004>     <1> csftdf_r_size:        resd 1
00018D80 <res 00000004>     <1> csftdf_w_size:        resd 1
00018D84 <res 00000004>     <1> csftdf_sf_rbytes:     resd 1
00018D88 <res 00000004>     <1> csftdf_df_wbytes:     resd 1
00018D8C <res 00000001>     <1> csftdf_percentage:    resb 1
                            <1> ; 17/03/2016
00018D8D <res 00000001>     <1> csftdf_videopage:     resb 1
00018D8E <res 00000002>     <1> csftdf_cursorpos:     resw 1
00018D90 <res 00000004>     <1> csftdf_sf_drv_dt:     resd 1
00018D94 <res 00000004>     <1> csftdf_df_drv_dt:     resd 1
                            <1>
                            <1> ; 21/03/2016
                            <1> ; 20/03/2016
                            <1> ; FILE.ASM
00018D98 <res 00000004>     <1> createfile_Name_Offset:  resd 1
00018D9C <res 00000004>     <1> createfile_FreeSectors:  resd 1
00018DA0 <res 00000004>     <1> createfile_size:         resd 1
00018DA4 <res 00000004>     <1> createfile_FFCluster:    resd 1 ; 11/03/2016
00018DA8 <res 00000004>     <1> createfile_LastDirCluster: resd 1
00018DAC <res 00000004>     <1> createfile_Cluster:      resd 1
00018DB0 <res 00000004>     <1> createfile_PCluster:     resd 1
00018DB4 <res 00000001>     <1> createfile_attrib:  resb 1
00018DB5 <res 00000001>     <1> createfile_SecPerClust:  resb 1
00018DB6 <res 00000002>     <1> createfile_DirIndex:     resw 1
00018DB8 <res 00000004>     <1> createfile_CCount:  resd 1
00018DBC <res 00000002>     <1> createfile_BytesPerSec:   resw 1 ; 23/03/2016
00018DBE <res 00000001>     <1> createfile_wfc:             resb 1
00018DBF <res 00000001>     <1> createfile_UpdatePDir:    resb 1 ; 31/03/2016
                            <1>
```

```
                                <1> ;alignb 4
                                <1>
                                <1> ; 11/04/2016
00018DC0 <res 00000002>         <1> env_var_length:       resw 1
                                <1>
00018DC2 <res 00000002>         <1> alignb 4
                                <1>
                                <1> ; 25/04/2016
00018DC4 <res 00000001>         <1> readi.valid: resb 1 ; valid data (>0 = valid for readi)
00018DC5 <res 00000001>         <1> readi.drv:   resb 1 ; drive number (0, 1,2,3,4..)
00018DC6 <res 00000001>         <1> readi.spc:   resb 1 ; sectors per cluster for 'readi' drive
00018DC7 <res 00000001>         <1> readi.s_index:  resb 1 ; sector index in current cluster (buffer)
00018DC8 <res 00000004>         <1> readi.sector:      resd 1 ; current disk sector
00018DCC <res 00000002>         <1> readi.bpc:   resw 1 ; bytes per cluster - 1
00018DCE <res 00000002>         <1> readi.offset:      resw 1 ; byte offset in cluster buffer
00018DD0 <res 00000004>         <1> readi.cluster:  resd 1 ; current cluster number
00018DD4 <res 00000004>         <1> readi.c_index:     resd 1 ; cluster index of the current cluster (0,1,2,3..)
00018DD8 <res 00000004>         <1> readi.fclust:      resd 1 ; first cluster of the current cluster
00018DDC <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>
00018DE0 <res 00000001>         <1> writei.valid:      resb 1 ; valid data (>0 = valid for writei)
00018DE1 <res 00000001>         <1> writei.drv:  resb 1 ; drive number (0, 1,2,3,4..)
00018DE2 <res 00000001>         <1> writei.spc:  resb 1 ; sectors per cluster for 'writei' drive
00018DE3 <res 00000001>         <1> writei.s_index: resb 1 ; sector index in current cluster (buffer)
00018DE4 <res 00000004>         <1> writei.sector:     resd 1 ; current disk sector
00018DE8 <res 00000002>         <1> writei.bpc:  resw 1 ; bytes per cluster - 1
00018DEA <res 00000002>         <1> writei.offset:     resw 1 ; byte offset in cluster buffer
00018DEC <res 00000004>         <1> writei.cluster: resd 1 ; current cluster number
00018DF0 <res 00000004>         <1> writei.c_index:    resd 1 ; cluster index of the current cluster (0,1,2,3..)
00018DF4 <res 00000004>         <1> writei.fclust:  resd 1 ; first cluster of the current cluster
00018DF8 <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
00018DFC <res 00000004>         <1> writei.lclust:     resd 1 ; writei last cluster (mget_w) ; 23/10/2016
00018E00 <res 00000004>         <1> writei.l_index:    resd 1 ; writei last cluster index (mget_w) ; 23/10/2016
00018E04 <res 00000001>         <1> writei.ofn:  resb 1 ; open file number (to be written) ; 23/10/2016
                                <1>
00018E05 <res 00000003>         <1> alignb 4
                                <1>
                                <1> ; 29/04/2016
00018E08 <res 00000004>         <1> Run_CDirFC:  resd 1
00018E0C <res 00000001>         <1> Run_Auto_Path:     resb 1
00018E0D <res 00000001>         <1> Run_Manual_Path: resb 1 ; 0 -> auto path sequence needed
00018E0E <res 00000001>         <1> EXE_ID:            resb 1
00018E0F <res 00000001>         <1> EXE_dot:    resb 1
                                <1>
                                <1> ; 06/05/2016
00018E10 <res 00000004>         <1> mainprog_return_addr: resd 1
00018E14 <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
00018E18 <res 00000004>         <1> video_eax:   resd 1  ; eax return value of video function
                                <1>
                                <1> ; 01/06/2016
00018E1C <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])
00018E20 <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')
00018E21 <res 00000001>         <1> p_change:    resb 1  ; process change status (for timer events)
                                <1> ; 23/05/2016 - TRDOS 386 ('clock')
00018E22 <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
00018E23 <res 00000001>         <1> timer_events:      resb 1  ; number of (active) timer events, <= 16
                                <1>
                                <1> ; 24/06/2016
00018E24 <res 00000001>         <1> w_str_cmd:   resb 1 ; WRITE STRING command (0,1,2,3) ; video.s
00018E25 <res 00000001>         <1> p_crt_mode:  resb 1  ; previous video mode (=3 or 0), backup mark/sign
                                <1> ; 26/06/2016
00018E26 <res 00000001>         <1> p_crt_page:  resb 1  ; previous active page (for 'set_mode')
                                <1> ; 04/07/2016
00018E27 <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
00018E28 <res 00000002>         <1> CRT_LEN:     resw 1  ; length of regen buffer in bytes
00018E2A <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)
00018E3A <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
00018E3E <res 00000001>         <1> VGA_MTYPE:   resb 1  ; 0=CTEXT,1=MTEXT,2=CGA,3=PLANAR1,4=PLANAR4,5=LINEAR
                                <1>
                                <1> ; 23/10/2016
00018E3F <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
00018E40 <res 00000004>         <1> FFF_UBuffer: resd 1  ; User's buffer address for FFF & FNF system calls
                                <1> ; 15/10/2016
00018E44 <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)
00018E45 <res 00000001>     <1> FFF_Attrib:  resb 1 ; Find First File attributes for Find Next File (LB)
00018E46 <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)
00018E47 <res 00000001>     <1> SWP_inv_fname:     resb 1 ; Set Working Path - Invalid File Name
00018E48 <res 00000002>     <1> SWP_Mode:    resw 1 ; Set Working Path - Mode
00018E4A <res 00000001>     <1> SWP_DRV:     resb 1 ; Set Working Path - Drive
00018E4B <res 00000001>     <1> SWP_DRV_chg: resb 1 ; Set Working Path - Drive Change
                            <1>
                            <1> ; 27/02/2017
00018E4C <res 00000001>     <1> fpready:     resb 1 ; '80387 fpu is ready' flag
                            <1>
                            <1> ; 08/10/2016
00018E4D <res 00000009>     <1> device_name:    resb 9  ; capitalized (and zero padded) device canem
                            <1>                     ; (example: "TTY0",0,0,0,0,0")
                            <1>
00018E56 <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)
00018E58 <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>
00018E78 <res 00000040>     <1> IDEV_NAME:  resb 8*NUMIDEV
                            <1>                     ; 8 byte names of installable device drivers
                            <1>
00018EB8 <res 00000008>     <1> IDEV_TYPE:  resb NUMIDEV ; Driver type of installable device drivers
00018EC0 <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.)
00018EC8 <res 00000020>     <1> IDEV_OADDR: resd NUMIDEV ; open function addr for installable dev driver
00018EE8 <res 00000020>     <1> IDEV_CADDR: resd NUMIDEV ; close function addr for installable dev driver
00018F08 <res 00000020>     <1> IDEV_RADDR: resd NUMIDEV ; read function addr for installable dev driver
00018F28 <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
00018F48 <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
00018F66 <res 0000001E>     <1> DEV_R_OWNER:  resb NUMOFDEVICES   ; Reading owner no (u.uid) of devices
00018F84 <res 0000001E>     <1> DEV_R_OPENCOUNT: resb NUMOFDEVICES   ; Reading open count
00018FA2 <res 0000001E>     <1> DEV_W_OWNER:  resb NUMOFDEVICES   ; Writing owner no (u.uid) of devices
00018FC0 <res 0000001E>     <1> DEV_W_OPENCOUNT: resb NUMOFDEVICES   ; Writing open count
00018FDE <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
00018FFC <res 0000001E>     <1> DEV_OPENMODE:       resb NUMOFDEVICES   ; 1 = read mode
                            <1>                                   ; 2 = write mode
                            <1>                                   ; 3 = read & write
```

```
                            <1>                                 ; 0 = not open (free)
0001901A <res 00000078>     <1> DEV_NAME_PTR:    resd NUMOFDEVICES   ; pointers to name addresses of drivers
                            <1>                                 ; Address base: KDEV_NAME+
                            <1>                                 ; or IDEV_NAME+
00019092 <res 00000078>     <1> DEV_R_POINTER:      resd NUMOFDEVICES   ; reading pointer, writing pointer
0001910A <res 00000078>     <1> DEV_W_POINTER:      resd NUMOFDEVICES   ; sector number if block device
                            <1>                                 ; character offset if char device
00019182 <res 00000002>     <1> alignb 4
                            <1>
                            <1> ; 06/10/2016
                            <1> ; Open File Parameters
00019184 <res 00000028>     <1> OF_FCLUSTER: resd OPENFILES  ; First clusters of open files
000191AC <res 0000000A>     <1> OF_DRIVE:    resb OPENFILES  ; Logical DOS drive numbers of open files
000191B6 <res 0000000A>     <1> OF_MODE:     resb OPENFILES  ; Open mode (1 = read, 2 = write, 3 = r&w)
000191C0 <res 0000000A>     <1> OF_STATUS:   resb OPENFILES  ; (bit 0 = read, bit 1 = write)
000191CA <res 0000000A>     <1> OF_OPENCOUNT:      resb OPENFILES  ; Open counts of open files
000191D4 <res 00000028>     <1> OF_POINTER:  resd OPENFILES      ; File seek/read/write pointer
000191FC <res 00000028>     <1> OF_SIZE:     resd OPENFILES      ; File sizes of open files (in bytes)
00019224 <res 00000028>     <1> OF_DIRFCLUSTER:    resd OPENFILES  ; Directory First Clusters of open files
0001924C <res 00000028>     <1> OF_DIRCLUSTER:     resd OPENFILES  ; Directory (Entry) Clusters of open files
00019274 <res 00000028>     <1> OF_VOLUMEID: resd OPENFILES  ; Vol ID for removable drives of open files
0001929C <res 00000028>     <1> OF_CCLUSTER: resd OPENFILES  ; Current clusters of open files
000192C4 <res 00000028>     <1> OF_CCINDEX:  resd OPENFILES  ; Cluster index numbers of current clusters
                            <1> ; 24/10/2016
000192EC <res 00000014>     <1> OF_DIRENTRY: resw OPENFILES  ; Directory entry index no. in dir cluster
                            <1>                             ; Sector index = entry index / 16
                            <1> ;alignb 2
                            <1>
00019300 <res 00000060>     <1> DTA:         resd 24            ; Find First File data transfer area
                            <1>
                            <1> ; 19/12/2016
00019360 <res 00000001>     <1> tcallback:   resb 1       ; Timer callback method flag for 'systimer'
00019361 <res 00000001>     <1> trtc:        resb 1       ; Timer interrupt type flag for 'systimer'
                            <1> ; 20/02/2017
00019362 <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>
00019363 <res 00000001>     <1> alignb 4
                            <1> ; 13/04/2017
00019364 <res 0000001E>     <1> DEV_INTR:   resb NUMOFDEVICES ; Device Interrupt (IRQ) number + 1
                            <1>                          ; (0= not available, 1= IRQ 0, 16= IRQ 15)
00019382 <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
000193C2 <res 00000009>     <1> IRQ.owner:   resb 9       ; owner, 0 = free, >0 = [u.uno]
000193CB <res 00000009>     <1> IRQ.dev:     resb 9       ; 0 = default/kernel, >0 = device number
000193D4 <res 00000009>     <1> IRQ.method:  resb 9             ; 0 = Signal Response Byte, 1 = Callback
000193DD <res 00000009>     <1> IRQ.srb:     resb 9             ; Signal Response/Return Byte value
000193E6 <res 00000024>     <1> IRQ.addr:    resd 9       ; Rignal Response Byte address (physical)
                            <1>                          ; or Callback service address (virtual)
                            <1> ; 28/02/2017
0001940A <res 00000004>     <1> IRQ_cr3:     resd 1       ; for saving cr3 register in IRQ handler
0001940E <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
0001940F <res 00000001>     <1> alignb 4
00019410 <res 00000001>     <1> audio_pci:   resb 1
00019411 <res 00000001>     <1> audio_device:      resb 1
00019412 <res 00000001>     <1> audio_mode:  resb 1
00019413 <res 00000001>     <1> audio_intr:  resb 1
00019414 <res 00000001>     <1> audio_busy:  resb 1  ; Busy flag for audio irq ; 21/04/2017
00019415 <res 00000001>     <1> audio_reserved: resb 1
00019416 <res 00000002>     <1> audio_io_base:     resw 1       ; Base I/O address of audio device
00019418 <res 00000004>     <1> audio_dev_id:      resd 1 ; BUS/DEV/FN ; 00000000BBBBBBBBDDDDDFFF00000000
0001941C <res 00000004>     <1> audio_vendor:      resd 1
00019420 <res 00000004>     <1> audio_stats_cmd: resd 1
                            <1> ;
00019424 <res 00000004>     <1> audio_buffer:      resd 1 ; virtual address of user's audio buffer
00019428 <res 00000004>     <1> audio_p_buffer:    resd 1 ; Physical address of user's audio buffer
0001942C <res 00000004>     <1> audio_buff_size: resd 1 ; user's audio buffer size (half buffer size)
00019430 <res 00000004>     <1> audio_dma_buff: resd 1  ; dma buffer address
00019434 <res 00000004>     <1> audio_dmabuff_size: resd 1 ; dma buffer size (2 * half buffer size)
00019438 <res 00000001>     <1> audio_flag:  resb 1  ; dma buffer flag (1st half = 0, 2nd half = 1)
00019439 <res 00000001>     <1> audio_user:  resb 1 ; user number of the owner
0001943A <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
0001943B <res 00000001>     <1> audio_srb:   resb 1 ; signal response byte value
0001943C <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>
00019440 <res 00000001>     <1> audio_bps:   resb 1  ; selected mode: 8 bit, 16 bit
00019441 <res 00000001>     <1> audio_stmo:  resb 1 ; selected mode: mono /stereo
00019442 <res 00000002>     <1> audio_freq:  resw 1 ; sampling rate
                            <1>
                            <1> ; 21/04/2017
00019444 <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)
00019445 <res 00000001>     <1> audio_flag_eol:    resb 1  ; End of Link status (vt8233, EOL/FLAG)
                            <1>
                            <1> audio_master_volume:
00019446 <res 00000001>     <1> audio_master_volume_l: resb 1 ; sound volume (lineout) left channel
00019447 <res 00000001>     <1> audio_master_volume_r: resb 1 ; sound volume (lineout) right channel
                            <1>
```

```
 734                                    <1> alignb 4
 735                                    <1> ; 28/05/2017
 736                                    <1> ; AC'97 Audio Controller Base Adress Registers
 737 00019448 <res 00000002>            <1> NAMBAR:             resw 1 ; Native Audio Mixer Base Address
 738 0001944A <res 00000002>            <1> NABMBAR:    resw 1 ; Native Audio Bus Mastering Base Address
 739                                    <1>
 740                                    <1> ;alignb 4
 741                                    <1> ; 21/04/2017
 742 0001944C <res 00000400>            <1> audio_bdl_buff:    resd 32*8  ; VT8233 (AC97) BDL Buffer Size
 743                                    <1> ; 12/05/2017
 744 0001984C <res 00000004>            <1> base_addr:  resd 1 ; 'direct_memory_access' (memory.s)
 745                                    <1>
 746                                    <1> ; 28/08/2017
 747                                    <1> ; 20/08/2017
 748 00019850 <res 00000001>            <1>             resb 1  ;
 749 00019851 <res 00000001>            <1> dma_user:   resb 1 ; user number for sysdma
 750 00019852 <res 00000001>            <1> dma_channel: resb 1 ; dma channel for sysdma
 751 00019853 <res 00000001>            <1> dma_mode:   resb 1  ; dma mode for sysdma
 752 00019854 <res 00000004>            <1> dma_addr:   resd 1 ; dma buffer physical addr for sysdma
 753 00019858 <res 00000004>            <1> dma_size:   resd 1  ; dma buffer size (in bytes) for sysdma
 754 0001985C <res 00000004>            <1> dma_start:  resd 1  ; dma start address for sysdma
 755 00019860 <res 00000004>            <1> dma_count:  resd 1  ; dma count (in bytes) for sysdma
 756                                    <1>
 757 00019864 <res 0000679C>            <1> alignb 65536
 758                                    <1> ; 09/08/2017
 759                                    <1> ; 12/05/2017
 760 00020000 <res 00010000>            <1> sb16_dma_buffer: resb 65536 ; DMA buffer for sb16 audio playing.

3636                                        ; 24/01/2016
3637                                        %include 'ubss.s'  ; UNINITIALIZED KERNEL (USER) DATA
   1                                    <1> ; ****************************************************************************
   2                                    <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.0) - UNINITIALIZED USER DATA : ubss.s
   3                                    <1> ; ----------------------------------------------------------------------------
   4                                    <1> ; Last Update: 28/02/2017
   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> ; ux.s (04/12/2015)
  12                                    <1> ; ****************************************************************************
  13                                    <1>
  14                                    <1> ; Retro UNIX 386 v1 Kernel - ux.s
  15                                    <1> ; Last Modification: 04/12/2015
  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> ; ----------------------------------------------------------------------------
  22                                    <1> ; Derived from UNIX Operating System (v1.0 for PDP-11)
  23                                    <1> ; (Original) Source Code by Ken Thompson (1971-1972)
  24                                    <1> ; <Bell Laboratories (17/3/1972)>
  25                                    <1> ; <Preliminary Release of UNIX Implementation Document>
  26                                    <1> ; (Section E10 (17/3/1972) - ux.s)
  27                                    <1> ; ****************************************************************************
  28                                    <1>
  29                                    <1> alignb 2
  30                                    <1>
  31                                    <1> inode:
  32                                    <1>       ; 11/03/2013.
  33                                    <1>       ;Derived from UNIX v1 source code 'inode' structure (ux).
  34                                    <1>       ;i.
  35                                    <1>
  36 00030000 <res 00000002>            <1>       i.flgs:       resw 1
  37 00030002 <res 00000001>            <1>       i.nlks:       resb 1
  38 00030003 <res 00000001>            <1>       i.uid:  resb 1
  39                                    <1>         ;i.size:  resw 1 ; size
  40 00030004 <res 00000002>            <1>       resw 1 ; 29/04/2016
  41 00030006 <res 00000010>            <1>       i.dskp:       resw 8 ; 16 bytes
  42 00030016 <res 00000004>            <1>       i.ctim:       resd 1
  43 0003001A <res 00000004>            <1>       i.mtim:       resd 1
  44 0003001E <res 00000002>            <1>       i.rsvd:  resw 1 ; Reserved (ZERO/Undefined word for UNIX v1.)
  45                                    <1>
  46                                    <1> I_SIZE      equ $ - inode
  47                                    <1>
  48                                    <1> process:
  49                                    <1>       ; 19/12/2016
  50                                    <1>       ; 21/05/2016
  51                                    <1>       ; 19/05/2016 - TRDOS 386 (TRDOS v2.0)
  52                                    <1>       ; 06/05/2015 - Retro UNIX 386 v1
  53                                    <1>       ; 11/03/2013 - 05/02/2014 (Retro UNIX 8086 v1)
  54                                    <1>       ;Derived from UNIX v1 source code 'proc' structure (ux).
  55                                    <1>       ;p.
  56                                    <1>
  57 00030020 <res 00000020>            <1>         p.pid:   resw nproc
  58 00030040 <res 00000020>            <1>         p.ppid:  resw nproc
  59 00030060 <res 00000020>            <1>         p.break: resw nproc
  60 00030080 <res 00000010>            <1>         p.ttyc:  resb nproc ; console tty in Retro UNIX 8086 v1.
  61 00030090 <res 00000010>            <1>       p.waitc: resb nproc ; waiting channel in Retro UNIX 8086 v1.
  62 000300A0 <res 00000010>            <1>       p.link:       resb nproc
  63 000300B0 <res 00000010>            <1>       p.stat:       resb nproc
  64                                    <1>
  65                                    <1>       ; 06/05/2015 (Retro UNIX 386 v1 feature only !)
  66 000300C0 <res 00000040>            <1>       p.upage: resd nproc ; Physical address of the process's
  67                                    <1>                           ; 'user' structure
  68                                    <1>       ; 21/05/2016
  69                                    <1>       ; 19/05/2016 (TRDOS 386 feature only!)
  70 00030100 <res 00000010>            <1>       p.timer: resb nproc ; number of timer events of the processs
  71                                    <1>
  72                                    <1>       ; 19/12/2016
  73 00030110 <res 00000040>            <1>       p.tcb: resd nproc ; timer callback service address (if > 0)
  74                                    <1>
  75                                    <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> ;        |                                          |
                        <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 requlat' ; 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
3638
00030459 <res 00000003>          alignb 4
3640
                                 ; 23/05/2016 (TRDOS 386)
                                 ; 14/10/2015 (Retro UNIX 386 v1, 'unix386.s')
0003045C <res 00000004>          cr3reg:      resd 1  ; cr3 register content at the beginning of the timer
                                              ; (or RTC) interrupt handler.
3645
                                 ; 10/12/2016 (callback)
                                 ; 10/06/2016
                                 ; 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)
3663
                                 ;; 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
```

```
3668 00030566 <res 00000004>          swpd_size:  resd 1 ; size of swap drive/disk (volume) in sectors (512 bytes).

3669 0003056A <res 00000004>          swpd_free:  resd 1 ; free page blocks (4096 bytes) on swap disk/drive (logical)
3670 0003056E <res 00000004>          swpd_next:  resd 1 ; next free page block
3671 00030572 <res 00000004>          swpd_last:  resd 1 ; last swap page block
3672
3673 00030576 <res 00000002>          alignb 4
3674
3675                                   ; 10/07/2015
3676                                   ; 28/08/2014
3677 00030578 <res 00000004>          error_code: resd 1
3678                                   ; 29/08/2014
3679 0003057C <res 00000004>          FaultOffset: resd 1
3680                                   ; 21/09/2015
3681 00030580 <res 00000004>          PF_Count:   resd 1 ; total page fault count
3682                                                     ; (for debugging - page fault analyze)
3683                                                     ; 'page_fault_handler' (memory.inc)
3684                                                     ; 'sysgeterr' (u9.s)
3685
3686                                   ; 29/04/2016 (TRDOS 386 = TRDOS v2.0)
3687                                   ; 22/08/2015 (Retro UNIX 386 v1)
3688                                   buffer:
3689 00030584 <res 00000008>                resb   8
3690                                   readi_buffer:
3691 0003058C <res 00000200>                resb   512
3692 0003078C <res 00000008>                resb   8
3693                                   writei_buffer:
3694 00030794 <res 00000200>                resb   512
3695                                   ; 24/10/2016
3696 00030994 <res 00000008>                resb   8
3697                                   rw_buffer:
3698 0003099C <res 00000800>                resb   2048  ; general purposed, r/w sector buffer
3699
3700                                   %if 1
3701                                   ; 17/01/2021
3702 0003119C <res 00000080>          edid_info:  resb 128 ; VESA EDID (monitor capabilities) info
3703                                   ; 28/11/2020
3704 0003121C <res 00000001>          pmi32:      resb 1 ; (>0) use VESA VBE3 protected mode calls
3705 0003121D <res 00000001>          vbe_mode_x: resb 1  ; VESA VBE3 video bios mode set options
3706 0003121E <res 00000002>          video_mode: resw 1 ; VESA VBE3 video mode (with option flags)
3707                                   ; 30/11/2020
3708 00031220 <res 00000004>          vbe3bios_addr: resd 1 ; new (writable mem) address of VBE3 bios
3709                                   ; 02/12/2020
3710 00031224 <res 00000004>          pmid_addr:  resd 1 ; PMInfoBlock ('PMID') linear address
3711                                   ; 14/01/2021
3712                                   ; 06/12/2020       ; VESA VBE 3 video state
3713                                   ;vbe3stbufsize: resw 1    ; video regs/dac/bios state buffer size
3714                                   ;                  ; block size in bytes
3715                                   ; 16/01/2021
3716 00031228 <res 00000002>          vbe3stbsflags: resw 1 ; video regs/dac/bios state buffer size
3717                                   ;                  ; pointer flags for buffer state options
3718                                   %endif
3719
3720                                   %if 1
3721                                   ; 10/12/2020
3722                                   LFB_Info:
3723 0003122A <res 00000010>                       resb 16 ; Linear Frame Buffer info block
3724
3725                                   ;24/11/2020 - TRDOS 386 v2.0.3
3726                                   ; BOCHS/PLEX86 VESA VBE3 MODE INFO extension to TRDOS 386 v2 kernel
3727                                   MODE_INFO_LIST:
3728 0003123A <res 00000044>                       resb 68 ; mode + 66 byte VESA vbe3 mode info (4F01h)
3729                                   %endif
3730
3731                                   ; 01/01/2021
3732 0003127E <res 00000004>          maskcolor:  resd 1 ; VGA/SVGA pixel mask color ('sysvideo')
3733                                   ; 05/01/2021
3734 00031282 <res 00000001>          ufont:      resb 1 ; (VGA graphics) user font flags
3735                                                     ; bit 7 - permission flag for int 31h
3736                                                     ; bit 4 - 8x16 user font ready/loaded flag
3737                                                     ; bit 3 - 8x8 user font ready/loaded flag
3738                                                     ; bit 1 - select 8x16 user font (sysvideo)
3739                                                     ; bit 0 - select 8x8 user font (sysvideo)
3740 00031283 <res 00000001>                       resb 1 ; 19/01/2021
3741                                   ; 17/01/2021
3742 00031284 <res 00000001>          srvsf:      resb 1 ; 'save restore video state' permission flag
3743                                   ; 18/01/2021
3744 00031285 <res 00000001>          srvso:      resb 1 ; video state buffer save/restore option
3745 00031286 <res 00000004>          VideoStateID: resd 1 ; used to verify state saved by same prog
3746
3747                                   bss_end:
3748
3749                                   ; 27/12/2013
3750                                   _end:  ; end of kernel code
```